ABSTRACT

Title of Dissertation:	DYNAMIC REPOSITIONING FOR BIKESHARING SYSTEMS
	Kiana Roshan Zamir, 2020
Dissertation directed by:	Professor, Ali Haghani Department of Civil and Environmental Engineering

Bikesharing systems' popularity has continuously been rising during the past years due to technological advancements. Managing and maintaining these emerging systems are indispensable parts of these systems and are necessary for their sustainable growth and successful implementation. One of the challenges that operators of these systems are facing is the uneven distribution of bikes due to users' activities. These imbalances in the system can result in a lack of bikes or docks and consequently cause user dissatisfaction.

A dynamic repositioning model that integrates prediction and routing is proposed to address this challenge. This operational model includes prediction, optimization, and simulation modules and can assist the operators of these systems in maintaining an effective system during peak periods with less number of unmet demands. It also can provide insights for planners by preparing development plans with the ultimate goal of more efficient systems.

Developing a reliable prediction module that has the ability to predict future station-level demands can help system operators cope with the rebalancing needs more effectively. In this research, we utilize the expressive power of neural networks for predicting station-level demands (number of pick-ups and drop-offs) of bikeshare systems over multiple future time intervals. We examine the possibility of improving predictions by taking into account new sources of information about these systems, namely membership type and status of stations.

A mathematical formulation is then developed for repositioning the bikes in the system with the goal of minimizing the number of unmet demands. The proposed module is a dynamic multi-period model with a rolling horizon which accounts for demands in the future time intervals. The performance of the optimization module and its assumptions are evaluated using discrete event simulation. Also, a three-step heuristic method is developed for solving large-size problems in a reasonable time. Finally, the integrated model is tested on several case studies from Capital Bikeshare, the District of Columbia's bikeshare program.

DYNAMIC REPOSITIONING FOR BIKESHARING SYSTEMS

by

Kiana Roshan Zamir

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park, in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2020

Advisory Committee:
Professor Ali Haghani, Chair
Professor Martin Dresner, Dean's Representative
Professor Gang-Len Chang
Associate Professor Barton Forman
Professor Paul Schonfeld

Dedication

This dissertation is dedicated to the memory of Arash Pourzarabi and Pouneh Gorji, my dear childhood friend and his lovely wife.

Acknowledgments

I would like to express my gratitude to my advisor, Professor Ali Haghani, for his guidance, kindness, patience, and support throughout the years. I am also grateful to the members of the advisory committee, Professor Martin Dresner, Professor Gang-Len Chang, Professor Barton Forman, and Professor Paul Schonfeld for their invaluable comments and suggestions. I would also like to extend my thanks to my internship mentor, Dr. Stefanie Brodie for her excellent guidance and encouragement during my internship at the District Department of Transportation. Furthermore, my sincere appreciation goes to all the Civil and Environmental Engineering Department staff and all the faculty at the University of Maryland (UMD) that I had the pleasure to attend their classes.

During my graduate studies, I had the privilege to work with and learn from many bright and incredible colleagues, co-authors, and friends: Arefeh Nasri, Iryna Bondarenko, Babak Baghaei, Mona Asudegi, Golnush Masghati, Moschoula Pternea, Binya Zhang, Zhongxiang Wang, Yeming Hao, Eirini Kastrouni, and the list goes on.

I am deeply thankful and blessed to meet and become friends with many amazing, kind, inspiring, and brilliant individuals at Maryland: Saba Ahmadi, Mahsa Derakhshan, Bahar Zarin, Melika Abolhassani, Saeed Seddigin, Amin Ghiasi, Mahyar Najibi, Soheil Behnezhad, Sanaz Aliari, Salimeh Gharazi, Afrooz Rahmati, Hadi Yami, Milad Gholami, Sina Dehghani, Ladan Najafi, Faezeh Dorri, Vahid Liaghat, and Reza Khani.

I am thankful to Ali Shafahi for being my best friend and biggest supporter during my studies at UMD. Without his support, this work would not have been possible. I would

also like to thank my lifelong friends and family members for being an incredible source of support when things were not that bright and encouraging: Mitra Rafiei, Majid Keshavarz, Behrooz Rafii, Roudabeh Moraghebi, Negin Alemazkoor, Majid Shafiee-jood, Haleh Sadat Ale Ahmad, Samira Aghaei, Mahshad Samnejad, Mahya Sadat Ghazizadeh Hashemi, Saman Bahadori, and Shahab Esmaeilnejad.

Most importantly, I wish to thank my parents, Soheila and Manouchehr for their endless love, support, sacrifice, and encouragement throughout my study, and my brother, Ashkan, for being the best big brother that anyone could ask for.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

1.1 *Backgrounds and Motivation*

Collaborative use of both goods and services has become very appealing with the recent technological advancements (Hamari, 2015). The rise in the popularity of sharing systems during the past years has caused different modes of sharing systems to emerge and become popular, especially in urban areas.

One of the most studied transportation-related sharing systems is vehicle sharing systems. Vehicle sharing programs (VSP) are defined in Nair (2010) as programs that "involve a fleet of vehicles located strategically at stations across the transportation network. Users are free to check out vehicles at any station and return the vehicles at stations close to their destinations (Nair, 2010)." The vehicle fleet can be cars, including electric vehicles, or bicycles. Among these two, the benefits of bikes in urban settings, where the distances are short, and the parking prices are high has caused the demand for such systems to increase. Bikesharing programs are ideal for short distance point-to-point trips in dense urban areas. They are also viable mobility options for the first/last mile of other public transport solutions (Nair, 2010; Toole design group, LLC and Foursquare ITP, 2013). Providing flexible mobility, travel time saving, reducing congestion and emission, decreasing car and fuel usage in addition to health benefits and support for multimodal transport connections are some other benefits of these systems (Shaheen, 2013; Fishman et al., 2013; Fishman, 2016).

More than 1000 cities had bikeshare systems in operation by July 2016 (DeMaio & Meddin, 2016). This number is eight times more than the number reported in 2010 (Shaheen et al., 2010). Figure 1-1 represents the growth in bikeshare cities from 1998 to 2013 (Fishman, 2016).

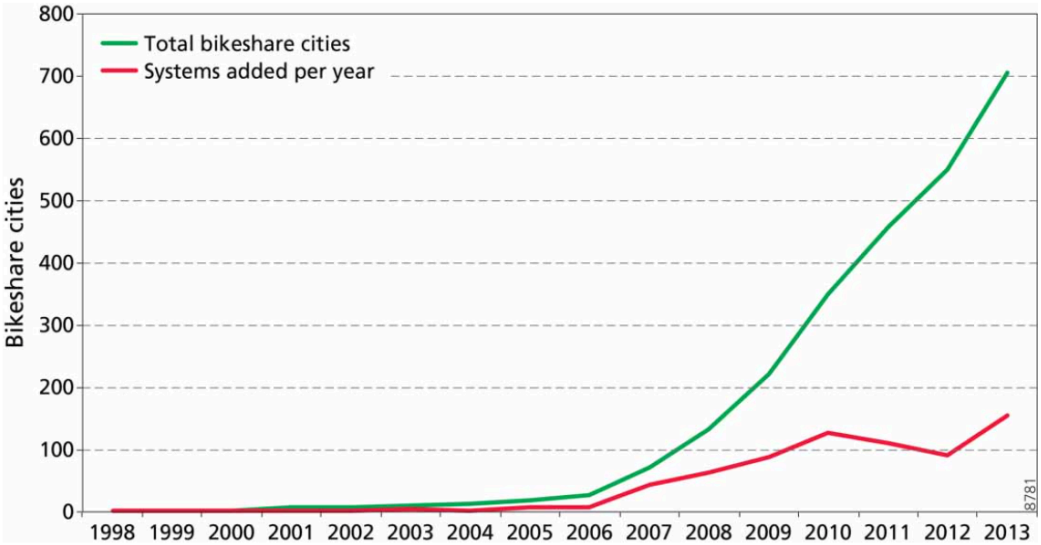


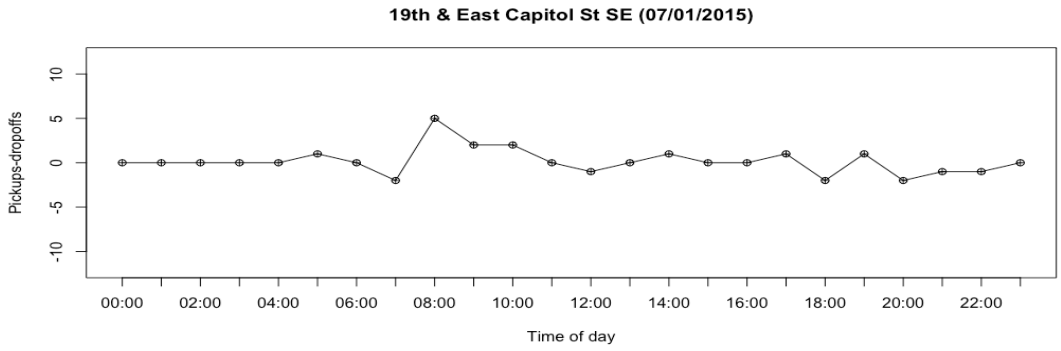
Figure 1-1 Growth in Bikeshare Cities (Fishman, 2016)

Maintaining these fast-growing systems is a difficult task. If the systems are not well-maintained, their users will become dissatisfied, and usage rates could decrease - jeopardizing the huge funds invested in expanding these systems. One major factor in measuring how well a bikeshare system is maintained is the availability of bikes and empty docks at stations when they are needed. The unavailability of bikes and docks is mainly a consequence of having an imbalanced system, which, as stated in Raviv & Kolka (2013) may have many reasons, such as:

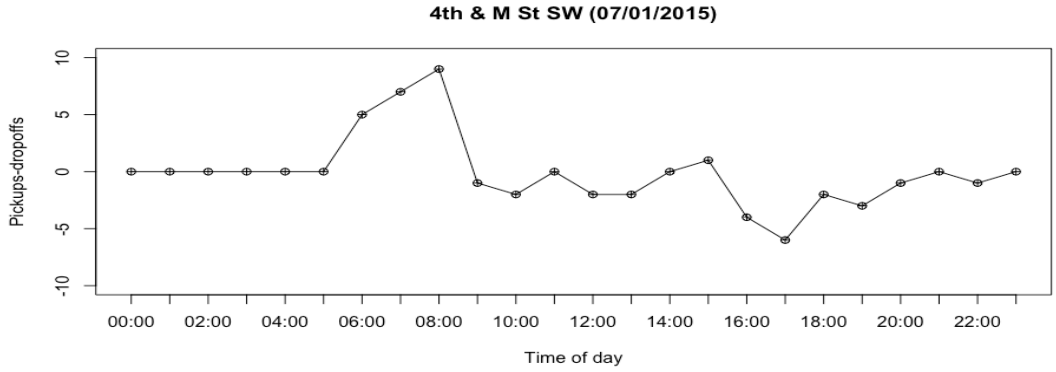
- Flow patterns,
- Availability and frequency of other transportation modes,
- The altitude of stations,

- Weather, and,
- Traffic conditions.

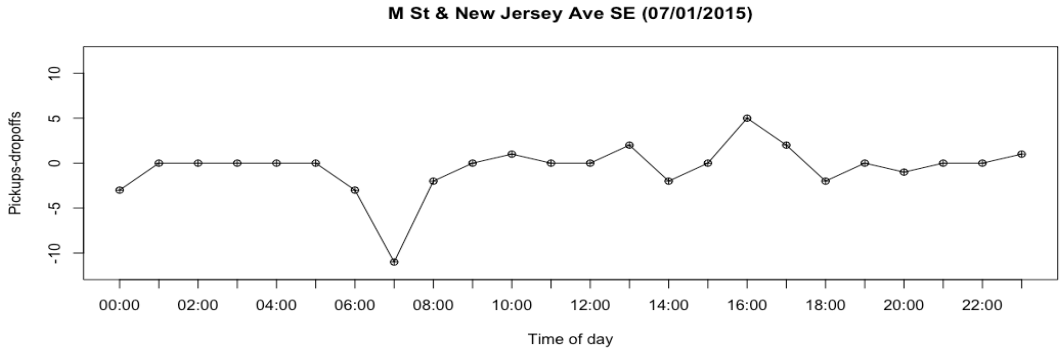
Figure 1-2 shows the difference between the number of pick-ups and drop-offs for three different stations that are part of the Capital Bikeshare program in the District of Columbia. Figure 1-2(a) represents a station that is relatively balanced throughout the day, whereas Figure 1-2(b) and Figure 1-2(c) represent stations that have significant gaps between their number of pick-ups and drop-offs. Figure 1-2(b) belongs to a station with a high number of pick-ups in the morning and a high number of drop-offs during the evening whereas Figure 1-2(c) belongs to a station with a high number of drop-offs in the morning and a high number of pick-ups in the evening. As Figure 1-2 suggests, to avoid having any unmet demands for pick-up (or drop-off), we need to provide bikes (or docks) for the difference between the number of pick-ups and drop-offs.



(a) 19th and East Capitol St SE



(b) 4th and M St SW



(c) M St and New Jersey Ave

Figure 1-2 Pick-ups and Drop-offs Profile for Three Stations

1.2 *Problem Statement, Research Objective, and Research Approach*

Since there is a limitation on the number of bikes and docks that can be added to the bikesharing systems, reducing unmet demand has been addressed in the literature using two approaches. The first one is by introducing a pricing mechanism that reduces these imbalances by imposing a reward to users who pick up their bikes from stations that have excess bikes or dropping off at stations that have a deficiency in the number of bikes. The second approach is by repositioning the bikes in the system. Fishman et al. (2013) describe this approach as "an operator moving bicycles across the network to maintain a more even distribution across the network (Fishman et al., 2013)."

The goal of this study is to develop an operational model for the dynamic repositioning of bikes of bikesharing systems with the objective of minimizing the number of unmet demands during peak hours. To achieve this, a Mixed Integer Programming (MIP) model is formulated. Also, a three-step heuristic method is introduced to reduce the solution time of the proposed model. A discrete event simulation module is incorporated into the proposed model to update the state of the system and to provide a framework for evaluating the operational models' assumptions as well as measuring their performance.

Similar to any operations research model, converting the bikesharing system's data into information can assist in modeling the structure of the system as well as finding optimal decisions for the system, and this can result in an increase in the effectiveness of the operation research model (Meisel, 2010). In the context of dynamic repositioning for bikesharing systems, predicting the demand for pick-ups and drop-offs for each station is a fundamental task, and a viable repositioning model should have an integrated

prediction and optimization module within. In this study, several models are compared in terms of performance, and the best model among them is chosen to evaluate the addition of two new features in enhancing the quality of predictions.

1.3 *Organization of the Dissertation*

The dissertation is organized as follows. Chapter 1 introduced the background and the motivation for this research. It also presented the problem statement, research objective, and research approach.

In Chapter 2, the literature on prediction models, along with studies focused on the prediction of the bikesharing systems' demand, are reviewed. In the context of operations research models, the literature on inventory optimization models and bikesharing routing models are discussed. This chapter finishes with a discussion on the research gap and expected contributions of this study.

Chapter 3 presents the proposed multi-period optimization model formulation for repositioning bikes in the bikeshare system, followed by a discussion on the simulation module used for simulating the bikeshare system.

Chapter 4 provides a detailed discussion of the data used for the case studies of this research, and Chapter 5 presents the results of the prediction module and analyzes the improvements in the performance in terms of root mean squared error resulting from introducing the status of stations and membership information.

Chapter 6 provides several numerical case studies for evaluating the assumptions of the optimization model proposed in chapter 3, followed by a sensitivity analysis on the solution time of the optimization model.

Chapter 7 presents the proposed heuristic algorithm to solve the multi-period optimization model, followed by comparing the solutions of the heuristic algorithm and the solutions of the model formulated in chapter 3. Also, the effect of the time interval is tested, and the quality of solutions are compared.

Chapter 8 presents the results of the implementation of the developed heuristic method on a large size case study. Finally, a summary, conclusions, and recommendations for future research are presented in Chapter 9.

Chapter 2: Literature Review

2.1 *Overview*

The concept of the bikeshare system was introduced in 1965 in Amsterdam and has evolved with the introduction and development of many technologies such as smartphones (DeMaio, 2009). Since the introduction of bikeshare systems, much research has been dedicated to developing descriptive and/or prescriptive models for different aspects of these systems.

Strategic level analyses, as well as operational level analyses, have been the focus of previous studies. Strategic level analyses include evaluating mobility patterns of potential users, demand analysis, finding the optimal location of stations and depot, and determining the total number of bikes needed for the system. The operational level analyses include service level analysis, analysis of the possible expansion of the system such as adding stations and/or docks (Nair, 2010; Schuijbroek et al., 2013; Erdoğan et al., 2014).

In the context of rebalancing operations that is the focus of this study, repositioning bikes throughout the systems is one of the widely used approaches around the world. Overall, avoiding lost demands by repositioning may involve many strategic *and* operational decisions, such as determining:

- The number of trucks needed,
- The number of bikes that need to be removed from or brought to each station,
- Allocation of trucks to stations,

- Routing of the trucks,
- The number of depots, and,
- The location of each depot.

Moreover, to reposition the bikes efficiently in the system, we need to have an estimate of the bikesharing system's various demands. As a result, developing demand/prediction models, along with repositioning models, are two of the most relevant research areas for addressing the repositioning problem.

2.2 *Prediction Models*

As stated above, a good repositioning plan requires accurate estimates of future demands. While having a good aggregate demand for the system is valuable for overall system evaluation, for the repositioning purpose, a more valuable prediction is more fine-grained and includes station level predictions. Nonetheless, most bikeshare demand predictions, due to their dependency and fluctuation over time, can be seen as cases of time-series predictions.

Here, we will briefly review some of the existing methodologies for forecasting time series data that have been applied in this study, as well as some of the studies that have worked on bikeshare demand estimation or prediction.

2.2.1 Dynamic Regression Models

Dynamic regression models (DRM) are one of the classical methods in the literature for modeling or predicting time series data. These models allow for the inclusion of both past observations and other information that may be relevant. Autoregressive

Integrated Moving Average (ARIMA) models and Autoregressive Moving Average (ARMA) models are two popular autoregressive models (Whittle, 1951). ARMA is a stationary model. For nonstationary models, one can use ARIMA, which achieves stationary by taking a series of differences. Equation (2.1) shows an example of a dynamic regression model with ARMA errors with an order of p for the autoregressive part and an order of q for the moving average part for predicting y_t. x_t is an additional information used for predicting y_t.

$$y_t = \beta x_t + \emptyset_1 y_{t-1} + \cdots + \emptyset_p y_{t-p} + \theta_1 z_{t-1} + \cdots + s\theta_q z_{t-q} + z_t \quad (2.1)$$

2.2.2 Decision Trees and Regression Trees

Classification and regression trees proposed by Brieman et al. (1984) are some of the widely used prediction models in the machine learning field, which can also be used for time series prediction. These models are developed by recursively partitioning the data space by a feature variable and establishing a structure that can be presented as a decision tree (Loh, 2011). Classification trees are used for predicting unordered dependent variables, and regression trees are for predicting ordered discrete or continuous dependent variables. Figure 2-1 illustrates a classification decision tree (Loh, 2011).

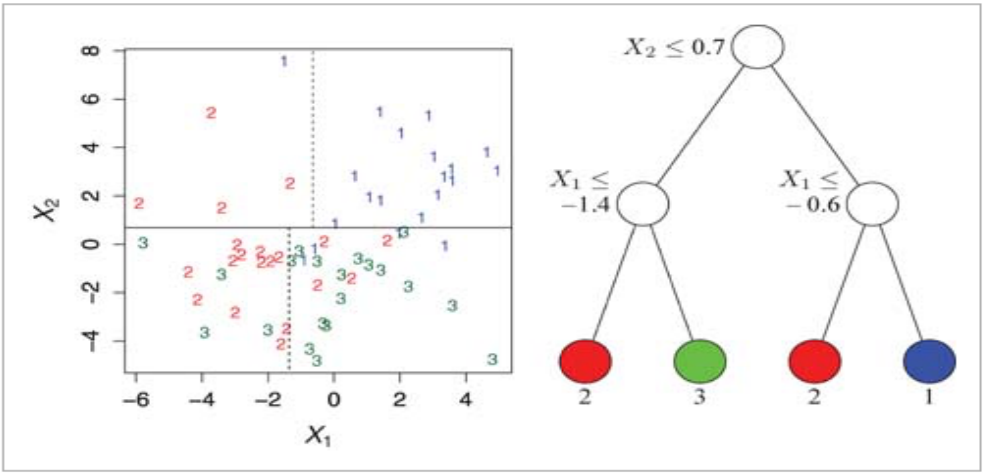


Figure 2-1 Classification Tree (Loh, 2011)

2.2.3 Neural Network Models

Neural network models are forecasting methods inspired by biological neural networks (McCulloch, 1943). These models usually have one input layer, several hidden layers, and one output layer. Input nodes are connected to the output layer via nodes in the hidden layers. One of the advantages of these models compared to dynamic regression models is that they are capable of modeling complex non-linear relationships between inputs and outputs. Some of the well-known variants of these models are:

- Multilayer Feedforward Neural Network:

As it is shown in Figure 2-2, the layout of a multilayer feedforward neural network is an acyclic directed graph consisting of several layers. Each layer has one or more simple processing elements, which are called neurons, and they are fully connected to the neurons that are in the next and previous layers. Neurons at each layer sum the weighted inputs, add a bias term and pass the result through an activation function. After determining the number of hidden layers and the number of neurons per layer,

the training set can be used for tuning the weights of the connections and the bias values. The weights are adjusted in a way that a loss function is minimized. A commonly used loss function for regression is the sum of squared errors between the outputs of the network and the true values. The back-propagation algorithm is mostly used for learning the weights (Safavian, 1991).

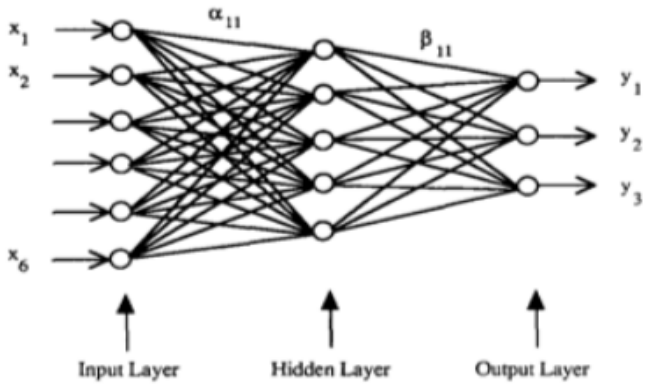


Figure 2-2 Multilayer Feedforward Network (Safavian, 1991)

- Recurrent Neural Network:

Recurrent Neural Networks (RNNs) are a class of artificial neural networks that have recurrent connections: they pass the output state of the neural network as the input of the next time step. This allows them to be able to model sequential data such as time-series of variable length and be able to represent previously seen states. In some sense, they have memories. One of the main differences of RNN type networks with feedforward networks is the variability in length of the input and output sequence, unlike feedforward networks where the architecture choice forces the input and output to be fixed to a special size and shape (Dupond, 2019). Similar to feedforward neural nets, RNNs are trained by back-propagation of the error signal throughout the neural

networks' weights. To be able to perform back-propagation, RNNs are "unrolled." Unrolling is the process of representing the RNN as a feedforward network by making multiple copies of the RNN such that each RNN cell passes the message (state) to its successor.

- Long Short-Term Memory:

Long Short-Term Memory (LSTM) networks are a special type of RNNs that have been developed by Hochreiter & Schmidhuber (1997) to solve the problem of long-term dependencies, which is related to vanishing/exploding gradients. LSTM network's building blocks are LSTM cells. While theoretically, standard RNNs should be capable of learning long-term dependencies, they often struggle a lot and require a lot of fine-tuning, as discussed in Bengio et al. (1994). LSTM's core idea is passing through the information of a cell's previous state to the next state with minor modifications. LSTM's have the ability to change the cell state through structures called gates carefully.

2.2.4 Ensemble Models

Ensemble models are a popular class of models in machine learning for improving classifications and predictions. These models are built based on training several models and then combining the result of these models to improve forecasting. Ensembling can be done in various ways, including:

- Bagging:

Short for bootstrap aggregating, was proposed by Breiman (1996). In this method, multiple predictors are combined to get an aggregated predictor. These predictors are often the same model that is trained on different subsets of data. These subsets are

selected at random with replacement from the entire training set. The aggregation strategy depends on the output task. If the task is regression and the predictor needs to predict continuous numerical values, the average of predictions by predictors can be used as the final prediction. If the task is classification, then the plurality (majority) vote can be used.

- Boosting:

Schapire (1990) developed a method for converting a weak learning algorithm (an algorithm that does slightly better than a random guess) into a highly accurate algorithm by combining the weak learning algorithms. This process is called boosting. One of the well-known boosting algorithms, AdaBoost, was developed by Freund & Schapire (1995), which is a classification algorithm based on combining multiple classification algorithms (called weak learners). AdaBoost adaptively resamples and reweights the training data to generate new weak learners, which improves the prediction for the misclassified instances.

- Random Forest:

Breiman (2001) proposed the idea of random forests for prediction. This method combines the idea of bagging and the concept of random feature selection, which has shown promising results in the work of Breiman (1996), Dietterich (1998), Ho (1998), and Amit & Geman (1997). A random forest classifier consists of a collection of tree-structured classifiers that are created by randomly selecting feature vectors for randomly selected training datasets.

- Stacking:

Wolpert (1992) proposed the idea of partitioning the training dataset into multiple parts and training based on one part of the training dataset and then correcting for biases based on the behavior of the model on the other part of the training data.

2.2.5 Prediction Models for Bikeshare Systems

Some of the prediction studies related to bikeshare systems focus on aggregate behavior predictions. Their focus is on estimating or identifying factors that affect the aggregated demand of bikeshare systems, such as the total number of bikes used per day or hour. Vogel & Mattfeld (2011) developed a regression model and an autoregressive regression model for predicting the total number of rentals using mostly weather data as the explanatory variable. Wang (2016) developed a random forest prediction model for predicting the total demand of the Citi Bike bikeshare system. Yin et al. (2012) applied four different models, namely: ridge linear regression, support vector regression (ε- SVR), random forest, and gradient boosted tree for predicting the hourly usage of the Capital Bikeshare system.

Station demand prediction is another line of research that has intensively been studied due to its relevance to rebalancing operations. Froehlich et al. (2009) provided a spatiotemporal analysis of bicycle usage. They developed a Bayesian network model for predicting the availability of bicycles for Barcelona's bikeshare system, "Bicing," Kaltenbrunner (2010) used an autoregressive moving average model that uses the information from surrounding stations as a predictor. Etienne & Latifa (2014) developed a model-based clustering methodology which groups the stations with similar bike usage, and then use this for predicting the demand of the clusters. Chen et

al. (2013) proposed a class of algorithms that use Generalized Additive Models for predicting bike availability. Li et al. (2015) proposed a hierarchical prediction model for predicting the rented and returned bikes from and to each station. Faghih-Imani et al. (2014) identified the determining factors of bikeshare usage for Montreal's bikeshare system, "BIXI" by developing models for the arrival and departure rates of stations. Yoon et al. (2012) developed a spatiotemporal prediction model for Dublin's bikeshare system, which can be used to help users of bikeshare systems in planning their trips. Giot & Cherrier (2014) examined various regressors for predicting the usage of bikeshare systems and concluded that by adding these regressors, the prediction models perform better than the baseline models. Faghih-Imani & Eluru (2016) developed a pooled spatial model with random effect, temporally lagged observed variables, and temporally and spatially lagged dependent variables for arrival and departure rates of Citi Bike bikeshare users.

2.3 *Repositioning Related Models*

Addressing the rebalancing problem by developing mathematical models for repositioning the bikes in the system is another stream of research related to bikesharing systems. We can divide the studies in this area into three major categories: Inventory level optimization, routing models, and combined models, which integrate the inventory level optimization and routing in the bikeshare systems.

2.3.1 Inventory Level Optimization

Some studies treat the problem of allocating bikes to stations (balancing) as an inventory problem. These studies only focus on determining the target inventory for

each station. Saltzman (2016) simulated the bikesharing system of San Francisco and tested several scenarios for improving the performance measurements of the bikeshare system.

Raviv & Kolka (2013) developed an inventory management model for bikesharing systems. They defined a dissatisfaction function as a function of initial inventory and found an approximation for this function. This function incorporates the expected penalty due to the abandonment of users including those who want to return the bike and those who want to rent a bike given that we have two independent non-homogeneous Poisson demand streams for the return's rate and renter's rate and the relative weights of dissatisfaction for unfulfilled requests. They proved this function is convex in the inventory level at the station at the beginning of the period and developed a method for estimating the function efficiently and providing bounds for their estimation. Kapsi et al. (2016) improved the work by accounting for unusable bikes.

Datner et al. (2019) developed a model for determining the initial inventory level of stations for daily repositioning. The model accounts for the spillover that may occur due to empty or full stations and the interaction among stations.

2.3.2 Routing

Studies that focus on the routing part of repositioning can be further classified into static models and dynamic models.

- Static Routing:

Static routing based models assume that the system remains static throughout the routing process. This assumption implies that bikes are not picked up or dropped off during the routing period. This assumption is mainly valid for night-time repositioning. In these studies, the target inventory is known, and the objective is to find the minimum cost route for repositioning the bikes to get to the targeted inventory level. By nature, this problem is similar to several problems in the literature:

1. One-commodity pick up and delivery problem: In One-commodity pick up and delivery traveling salesman problem (1-PDTSP), a single vehicle with fixed capacity must either pick up or deliver known amounts of a single commodity to a given list of customers. There are two kinds of customers: delivery customers who want an amount of the product to be shipped to them and pick-up customers who want to provide an amount of the product. In 1- PDTSP, it is assumed that the product collected from the pick-up customers can be supplied to the delivery customers and that the initial load of the vehicle leaving the depot can be any quantity (Hernández-Pérez & Salazar-González, 2007). In the case of multiple vehicles, the problem is similar to one commodity pick up and delivery vehicle routing problem (1-VRPPD).
2. Swapping problem: In which, a vehicle with unit capacity wants to ship items between the vertices (Anily & Hassin, 1992). The objective is to minimize the route taken by the truck for accomplishing all the shipments. In the preemptive version of this problem, items can be dropped temporarily at vertices that are not their final destination, and later on, they can be picked up and moved to their final destinations. In the non-preemptive version, this is not possible.

Routing problems related to rebalancing the bikeshare systems can have different objective functions such as minimizing the total unmet demand and minimizing the total user dissatisfaction (Raviv et al., 2013), minimizing the sum of relocation and lost user costs (Forma et al., 2015), minimizing the total travel cost or routing time (Benchimol et al., 2011; Chemla et al., 2013; Dell'Amico et al. 2014; Dell'Amico et al., 2016; Erdoğan et al., 2014; Erdoğan et al., 2015), minimizing the deviation from the targeted number of bikes in each station (Ho & Szeto, 2014), minimizing the number of loading and unloading quantities (Papazek et al., 2013), minimizing the total travel time on all routes, and minimizing the maximum tour length (Schuijbroek et al., 2013) have been used as the objective function for the routing problem.

The maximum time for each repositioning activity, routing constraints requiring each node to be visited once, perfect balance, road condition, traffic regulations, geographical factors, and probabilistic level of service are some of the operational constraints considered in the past studies (Ho & Szeto, 2014).

The majority of the following studies develop deterministic models, which according to Shu et al. (2013), a deterministic model could be used for modeling these systems contrary to the stochastic nature of these systems.

Benchimol et al. (2011) proposed a 9.5 approximation algorithm based on an algorithm developed by Chalasani & Motwani (1999) (C-delivery traveling salesman problem) for changing the current state of stations to the desired state by using a single truck. Their objective was minimizing the moving cost.

Ho & Szeto (2014) solved a static repositioning problem with a single vehicle using iterative tabu search. Their objective function was minimizing the penalty cost of deviation of the number of bikes at each station after repositioning from the optimal number of bikes.

Raviv et al. (2013) proposed four different formulations: arc-indexed, time-indexed, sequence-indexed, and swapping-based for solving the repositioning problem. Minimizing user's dissatisfaction was used as the objective, and multiple vehicles were used for balancing the system.

Chemla et al. (2013) developed a MIP model for solving the single-vehicle repositioning problem with the objective of minimizing the cost of routing. They relaxed the constraint of visiting each station once and tried different branch and cut algorithms for reducing the computation time and used tabu search for finding an upper bound of the optimal solution.

Dell'Amico et al. (2014) presented four MIP formulations for solving the multiple vehicle rebalancing problems with the objective of minimizing the cost of routing. They tried different branch and cut algorithms to solve the problem. Dell'Amico et al. (2016) proposed a destroy and repair metaheuristic and compared their results with the branch and cut algorithm developed in their 2014 work.

Erdoğan et al. (2014) developed a static bicycle relocation model with demand intervals. In this problem, they redistributed bicycles among the system with a single capacitated vehicle, given that there are constraints on the lower and upper bounds of the required number of bikes at each station. Their objective was minimizing the

routing time. Branch-and-cut algorithm and benders decomposition were proposed for solving the problem.

Erdoğan et al. (2015) presented an exact algorithm for determining the minimum cost route for a single truck to pick up and drop-off the targeted number of bikes from the stations. Multiple visits and temporary storage were allowed.

Forma et al. (2015) proposed a 3-step heuristic approach with the objective of minimizing a combined function of station level penalties and total travel cost of vehicles. In the first step, stations are clustered based on their location and inventory considerations. The second step is to route the vehicles through the clusters and making decisions for each station independently. The final step is to solve the problem for all the stations. The traversal of vehicles is only allowed between two adjacent clusters.

Papazek et al. (2013) used greedy construction heuristic (GCH) and preferred iterative look ahead technique (PILOT) for finding initial solutions to the static balancing with the objective of minimizing the routing time, deviation of stations from their targeted inventory and the number of loading activities. Then they used variable neighborhood descent (VND) for improving these candidate solutions and then used a greedy randomized adaptive search procedure (GRASP) for returning the best solution.

Rainer-Harbach et al. (2013) used variable neighborhood search (VNS), and variable neighborhood descent (VND) for generating routes and they proposed three different approaches for optimal loading instruction given that the route is fixed. Greedy heuristic, maximum flow approach, and the linear programming approach were used.

Raidl et al. (2013) proposed a new method for calculating the load instructions, which is based on the maximum flow approach.

- Dynamic Routing:

In the beforementioned routing studies, the assumption was that the system is static, meaning the number of bikes required by, and present in each station is fixed. This assumption is usually true during nighttime repositioning in which the system is nearly idle. During the daytime operation, this assumption is not valid. Some studies tried to address the repositioning problem in the dynamic case. In the dynamic case, the number of bikes required by, and present in each station is changing over time.

Contardo et al. (2012) formulated the problem on a space-time network with the objective of minimizing the unmet demand. Then they used Dantzig-Wolf decomposition and Benders decomposition to solve the problem.

Vogel et al. (2014) proposed a model for minimizing the expected cost of relocation and violating the service level constraints. They used a hybrid metaheuristic integrating large neighborhood search with exact solution methods.

Chemla et al. (2013) proposed three heuristics for the dynamic repositioning of bicycles using one vehicle. They also proposed a pricing strategy for balancing the system and tested their methods using simulation.

Kloimüllner et al. (2014) modeled the dynamic case and used a greedy and PILOT heuristic, variable neighborhood search, and GRASP to solve the problem efficiently.

Wang (2014) developed two heuristic methods for solving dynamic repositioning problems using a single vehicle with the objective of minimizing the unmet demand and routing cost. The greedy algorithm and the rolling horizon approach were used to solve the problem. Benders' decomposition was used as an exact method for solving the problem.

Pfrommer et al. (2014) developed a dynamic routing algorithm and an incentive scheme with the objective of maximizing the number of bike trips.

Ghosh et al. (2016) used a robust repositioning approach by mimicking the system as an iterative two-player game, which assumed the environment could generate a worst demand scenario.

Ghosh et al. (2017) developed a dynamic repositioning model that accounts for the future expected demand. They used Lagrangian dual decomposition and clustering to solve the problem.

2.3.3 Combined Inventory and Routing Optimization

O'Mahony & Shmoys (2015) developed an optimization model for the desired filling level. They used two different strategies for routing the vehicles during the overnight balancing and mid-rush balancing.

Schuijbroek et al. (2013) used a cluster-first route-second approach in which they first converted the multi-vehicle rebalancing problem into a single-vehicle problem based on maximum spanning star approximation and then tried to minimize the maximum routing time of each vehicle while satisfying service level feasibility constraints.

Liu et al. (2016) developed prediction models for pick-ups and drop-offs of the NYC Citi Bikeshare system. Based on these two models, they determined the inventory target for each station. They developed two algorithms for clustering the stations and then solved the routing problem within the clusters.

2.4 *Contributions*

Reviewing the literature shows there are limited studies in the literature evaluating and combining prediction and optimization models for repositioning of bikes in the bikesharing systems (e.g., Liu et al., 2016). This study aims to further explore this line of research by:

- Improving station-level prediction of the number of pick-ups and drop-offs by including two new features -- membership information and status of stations -- into the prediction module of the repositioning model. Several models are compared as candidate underlying models for evaluating the additional value of introducing these new features/variables into the model. Feedforward neural networks are trained and used as the best model among the tested models, and results are presented using this method.
- The sensitivity of the predictions is tested with respect to two factors: prediction interval (i.e., the prediction time duration (e.g., 30 minutes)), and dynamic setting. Dynamic setting refers to using streaming data for predicting the number of pick-ups and drop-offs for the future interval instead of using a static pre-trained prediction model.

- A novel MIP model is formulated. The introduced model is a multi-period optimization model with a rolling horizon. The benefit of using this model is that it can improve the efficiency of the repositioning operations by accounting for the short term as well as long term changes in the state of the bikeshare system. This benefit is verified via numerical experiments in section 6.4. This approach has also been used by Ghosh et al. (2017). However, their model objective is to maximize the revenue and uses the movement of customers in the system as an input. Our proposed model uses the number of pick-ups and drop-offs from the stations as an input, which is more aggregated and considerably easier to predict.
- A discrete event simulation module is incorporated into the proposed framework. The simulation module will update the state of the system after each period. It will provide the ability to evaluate the optimization and prediction modules' assumptions as well as measuring their performance.
- A three-step heuristic method is introduced to reduce the solution time of the proposed model, and the quality of solutions are tested. The efficiency of the repositioning is tested with respect to interval duration using this heuristic method.

Chapter 3: Optimization Module and Simulation Module

This section proposes a MIP formulation for the proposed rebalancing problem. It also demonstrates the simulation module framework that can be used for validating modeling assumptions and evaluating different models.

3.1 *Optimization Module*

For the optimization module, a dynamic multi-period model is proposed. The advantages of the proposed multi-period model over some of the existing models in the literature are as follows. A multi-period model can account for the short-term demand as well as long-term demand. This feature is extremely important for repositioning bikes in the bikeshare systems. By planning for a specific interval, if the interval is short, the repositioning plan will not be efficient (i.e., some stations may need to be revisited during different intervals.) If the interval is long, we may miss some of the demands due to temporal changes in the system. For instance, if a station has high pick-ups for half of the interval and high drop-offs for the rest of the interval, the model may suggest we do not need to pick up or drop-off any bikes.

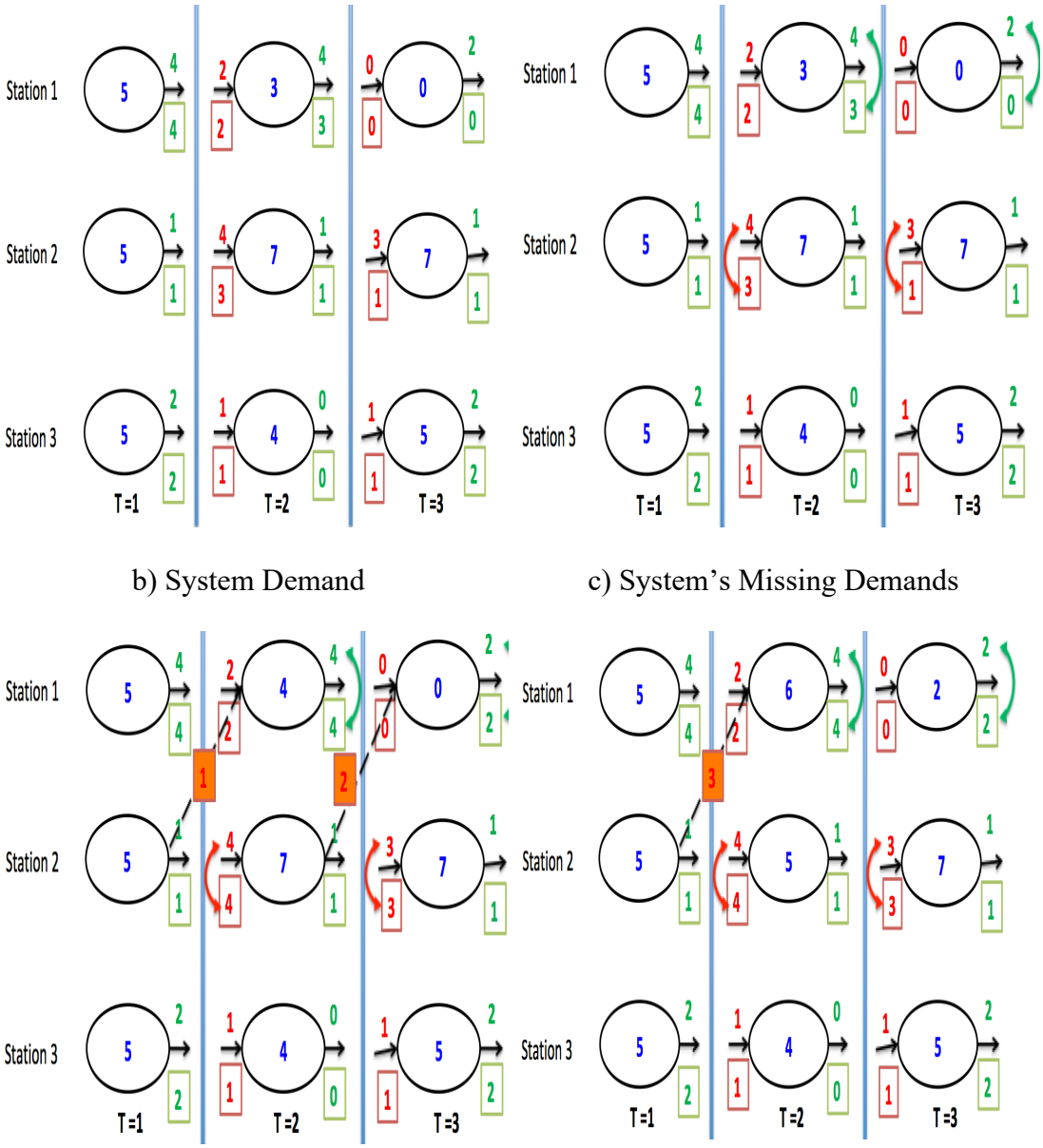
Another advantage of the proposed model is that since the model is dynamic, the repositioning can be adjusted after having new/updated information about future demand (i.e., when predictions are updated.) An example is provided in Figure 3-1 to demonstrate the benefits of this model further.

Figure 3-1(a) shows a system with three stations during two consecutive time intervals. All the stations have a capacity of seven bikes with the initial number of bikes equal to five. The number of bikes at the beginning of each interval is provided with a blue

color. Numbers in green show the potential demand for pick-ups, and Numbers in red show the potential demand for drop-offs. Numbers in the green box are the actual number of pick-ups, and numbers in the red box are the actual number of drop-offs (Figure 3-1(a)). Figure 3-1(c) shows the locations where the potential demands and the actual demands do not match. For instance, during the first interval, station 2 will have one potential pick-up and four potential drop-offs. Even in the optimistic scenario (i.e., one potential pick-up happens before the four potential drop-offs), the station can only serve three of the drop-offs. A truck can be used to move the bikes between the stations to serve all demands. Here, we assume that we know the pick-up and drop-off demand of each station for each time interval. Two different repositioning plans are proposed in Figure 3-1(d) and Figure 3-1(e). Repositioning 1 only looks at the next interval for planning the movements. As a result, we need two movements to serve all the demands during the two intervals. Repositioning 2 looks at the next two intervals. In this case, we only need one movement. As it is shown, repositioning 2 is more efficient compared to repositioning 1. We note that this analysis is only true if the predictions are close to the future demand.

Numbers in green = Potential number of pick-ups
Numbers in red = Potential number of drop-offs
Numbers in green with a box = Actual number of pick-ups
Numbers in red with a red box= Actual number of drop-offs
Blue color numbers = Number of bikes available at the station
Capacity of each station = 7

a) Legend



b) System Demand

c) System's Missing Demands

d) Repositioning 1

e) Repositioning 2

Figure 3-1 Sample Example

3.1.1 Model Assumptions

To formulate the problem, we have made two major assumptions. The first assumption is that an actual pick-up during a period adds extra capacity for drop-offs during the same period. Also, an actual drop-off during a period adds extra capacity for pick-up during that period. This assumption by itself follows from assuming that the pick-ups and drop-offs happen uniformly during each interval. The first assumption implies that a station will have unmet pick-up demand during a period only if the number of bikes available at the station at the beginning of the period plus the number of drop-offs is less than the pick-up demand. A similar statement is also true for the unmet drop-off demand. This assumption is optimistic; however, it should usually hold especially for short intervals. A simulation module is developed for the bikeshare system to examine this assumption.

The second major assumption is that the bikes that are going to be picked up for movement by trucks are not available for pick-up during the interval for both origin and destination station. We also need to have racks available for them at the destination stations for the entire repositioning interval. In contrast to the previous assumption, this is a conservative assumption. However, since the routing schedules are not finalized, it is logical to make this assumption. Not having this assumption may result in a discrepancy between loading plans and the availability of bikes or racks at the stations.

3.2 *Model Formulation*

3.2.1 Notation

This section provides the notation used for defining the sets, parameters, and decision variables as well as their description and their domain for the proposed optimization module.

- Sets

T	Set of time intervals/steps
S	Set of stations
K	Set of trucks
N	Set of stations + origin/destination node (artificial) $N = S \cup \{0\}$

- Parameters

P_s^t	Number of pick-ups at the station (s) at a time interval (t)
D_s^t	Number of drop-offs at the station (s) at a time interval (t)
$TT_{ss'}^t$	Travel time between station (s) and (s') at a time interval (t)
R_s	Capacity (Number of docks) of station (s)
A_s^0	Number of bikes at the station (s) at the starting time
C	Capacity of trucks
α,β	The weighting factor for the objective function
SS_k	Start station for the truck (k)
I	Duration of each interval

- Decision Variables

a_s^t Number of available bikes in the station (s) at the beginning of time interval (t)

ap_s^t Number of actual pick-ups at a time interval (t) for the station (s)

ad_s^t Number of actual drop-offs at a time interval (t) for the station (s)

bpp_s^t Binary indicator of unmet pick-up demand at a time interval (t) for the station (s)

bdp_s^t Binary indicator of unmet drop-off demand at a time interval (t) for the station (s)

pp_s^t Number of unmet pick-up demands at a time interval (t) for the station (s)

dp_s^t Number of unmet drop-off demand at a time interval (t) for the station (s)

q_{sk}^t Number of bikes picked from the station (s) by truck (k) at a time interval (t)

q'^t_{sk} Number of bikes dropped at the station (s) by truck (k) at a time interval (t)

bq_{sk}^t Binary indicator of bikes picked from the station (s) by truck (k) at a time interval (t)

bq'^t_{sk} Binary indicator of bikes dropped at the station (s) by truck (k) at a time interval (t)

$qtot\ ^t_{sk}$ Number of bikes in the truck (k) after visiting station (s) at a time interval (t)

$x_{ss'k}^t$ Binary indicator for the truck (k) traveling from the station (s) to station (s') at a time interval (t)

b_{sk}^t Binary indicator for truck (k) visiting station (s) at a time interval (t)

y_{sk}^{t} Auxiliary variable for subtour elimination purpose

$aux1_{s}^{t}$ Auxiliary variable for linearizing

$aux2_{s}^{t}$ Auxiliary variable for linearizing

$baux1_{s}^{t}$ Binary auxiliary variable for linearizing

$baux2_{s}^{t}$ Binary auxiliary variable for linearizing

3.2.2 Objective Function

Equation (3.1) shows the objective function of the model. The objective function has three parts. The first part is the lost demand penalty (number of unmet demands), which includes unmet pick-up demand and unmet drop-off demand. The second part of the objective function is the total routing time, and the third part is the initial penalty for moving bikes from or to a station. These terms' contributions to the overall objective are weighed using coefficients (α, β). These coefficients can be set based on the needs of the system and the operators' view. Here, we have assumed the operator's main goal is to minimize the unmet demand, so the second and third parts of the objective function have small coefficients to only avoid moving bikes in the system without any reduction in the unmet demand.

$$Minimize\ Z = \sum_{t\in T}\sum_{s\in S}(pp_{s}^{t} + dp_{s}^{t}) + \alpha * \sum_{t\in T}\sum_{k\in K}\sum_{s\in S}\sum_{s'\in S/\{s\}} TT_{ss'}^{t} \cdot x_{ss'k}^{t} + \beta * \sum_{t\in T}\sum_{k\in K}\sum_{s\in S} q_{sk}^{t} \quad (3.1)$$

3.2.3 Constraints

For each interval, the start and end location of all the trucks during each interval are set to be an artificial origin. For the first interval, each truck is visiting a specific node after the artificial origin defined by the parameter SS_k. This specific node is the actual location of the truck at the beginning of the planning problem. Constraints (3.2) to (3.4) make sure these constraints are met.

$$\sum_s x_{0sk}^t = 1 \qquad \forall t \in T, \forall k \in K \qquad (3.2)$$

$$\sum_s x_{s0k}^t = 1 \qquad \forall t \in T, \forall k \in K \qquad (3.3)$$

$$x_{0SS_k k}^1 = 1 \qquad \forall k \in K \qquad (3.4)$$

Constraints (3.5) ensure the first visited node at each interval is the last visited node at the end of the previous interval.

$$x_{0sk}^t = x_{s0k}^{t-1} \qquad \forall t \in T - \{1\}, \quad \forall k \in K, \forall s \in S \qquad (3.5)$$

Constraints (3.6) are conservation of flow constraints.

$$\sum_{s\prime \in N} x_{ss\prime k}^t = \sum_{s\prime \in N} x_{s\prime sk}^t \qquad \forall t \in T, \quad \forall k \in K, \forall s \in N \qquad (3.6)$$

Constraints (3.7) and (3.8) ensure that we cannot move any bikes or add bikes to a station if that station does not have enough bikes for pick up or enough rack for drop-off, respectively.

$$\sum_{k \in K} q_{sk}^{t} \leq (1 - \ bpp_s^t) * R_s \qquad \forall\, t \ \in T, s \in S \qquad (3.7)$$

$$\sum_{k \in K} q'^{t}_{sk} \leq (1 - \ bdp_s^t) * R_s \qquad \forall\, t \ \in T, s \in S \qquad (3.8)$$

As mentioned in section 3.1.1, the model assumption is that the bikes that are going to be picked for movement by trucks are not available for pick up during the interval for both origin and destination stations. Also, we need to have docks available for the bikes in the destined station for the entire repositioning interval. Constraints (3.9) and (3.10) enforce these constraints.

$$\sum_{k \in K} q_{sk}^{t} \leq max(0, a_s^t - \ ap_s^t) \qquad \forall\, t \in T, s \in S \qquad (3.9)$$

$$\sum_{k \in K} q'^{t}_{sk} \leq \max(0, R_s - a_s^t - \ ad_s^t) \qquad \forall\, t \in T, s \in S \qquad (3.10)$$

One set of continuous and one set of binary auxiliary variables are introduced to the model formulation and constraints (3.11) to (3.18) are added to the model to linearize constraints (3.9) and (3.10).

$$\sum_{k \in K} q_{sk}^{t} \leq aux1_s^t \qquad \forall\, t \in T, s \in S \qquad (3.11)$$

$$a_s^t - ap_s^t + M * R_s * baux1_s^t \geq 0 \qquad \forall t \in T, s \in S \qquad (3.12)$$

$$aux1_s^t \leq M * R_s * (1 - baux1_s^t) \qquad \forall t \in T, s \in S \qquad (3.13)$$

$$aux1_s^t \leq a_s^t - ap_s^t + M * R_s * baux1_s^t \qquad \forall t \in T, s \in S \qquad (3.14)$$

$$\sum_{k \in K} q'^t_{sk} \leq aux2_s^t \qquad \forall t \in T, s \in S \qquad (3.15)$$

$$R_s - a_s^t - ad_s^t + M * R_s * baux2_s^t \geq 0 \qquad \forall t \in T, s \in S \qquad (3.16)$$

$$aux2_s^t \leq M * R_s * (1 - baux2_s^t) \qquad \forall t \in T, s \in S \qquad (3.17)$$

$$aux2_s^t \leq R_s - a_s^t - ad_s^t + M * R_s * baux2_s^t \qquad \forall t \in T, s \in S \qquad (3.18)$$

Constraints (3.19) and (3.20) limit the number of actual pick-ups and drop-offs.

$$ap_s^t \leq ad_s^t + a_s^t - \sum_{k \in K} q_{sk}^t \qquad \forall t \in T, s \in S \qquad (3.19)$$

$$ad_s^t \leq R_s - a_s^t + ap_s^t - \sum_{k \in K} q_{sk}^t \qquad \forall t \in T, s \in S \qquad (3.20)$$

Constraints (3.21) relate the available number of bikes at station *s* at time *t+1* (a_s^{t+1}) to the available number of bikes at station *s* at time *t* (a_s^t) and the number of actual pick-ups (ap_s^t) and drop-off (ad_s^t) at the station during the interval (t) and the number

of bikes moved from other stations ($\sum_k q'^t_{sk}$) to this station, and the number of bikes moved from this station to other stations ($\sum_k q^t_{sk}$).

$$a_s^{t+1} = a_s^t - ap_s^t + ad_s^t + \sum_{k \in K} q'^t_{sk} - \sum_{k \in K} q^t_{sk} \qquad \forall t \in T, s \in S \qquad (3.21)$$

The number of bikes picked from stations is equal to the number of bikes dropped at stations (Constraints (3.22)).

$$\sum_{s \in S} q'^t_{sk} = \sum_{s \in S} q^t_{sk} \qquad \forall t \in T, \forall k \in K \qquad (3.22)$$

Constraints (3.23) are the truck load constraints.

$$q^t_{sk} \leq qtot^t_{sk} \leq C - q'^t_{sk} \qquad \forall t \in T, \forall k \in K, s \in S \qquad (3.23)$$

$$q'^t_{sk} \leq qtot^t_{s'k} + C * (1 - x^t_{s'sk}) \qquad \forall t \in T, \forall k \in K, s \in S \qquad (3.24)$$

Constraints (3.25) are conservation of load after visiting nodes which are linearized in Constraints (3.26).

$$qtot^t_{sk} \geq \left(qtot^t_{s'k} + q^t_{sk} - q'^t_{sk}\right) * x^t_{s'sk} \qquad \forall t \in T, \forall k \in K, \{s, s'\} \in S \qquad (3.25)$$

$$qtot_{sk}^{t} \geq qtot_{s'k}^{t} + q_{sk}^{t} - q'^{t}_{sk} - C * (1 - x_{s'sk}^{t}) \qquad \forall t \in T, \forall k \in K, \{s, s'\} \in S \qquad (3.26)$$

Travel time for each truck and during each interval should be less than interval duration. Loading and unloading time is not included in travel time calculations in this formulation. However, this constraint can be adjusted to include this term as well.

$$\sum_{s\epsilon S, s'\epsilon S/\{s\}} TT_{ss'}^{t} * x_{s'sk}^{t} \leq I \qquad \forall t \in T, \forall k \in K \qquad (3.27)$$

Constraints (3.28) set the initial number of bikes at the station.

$$a_s^1 = A_s^0 \qquad \forall s \in S \qquad (3.28)$$

Constraints (3.29) ensure that the station capacities are not violated.

$$a_s^t \leq R_s \qquad \forall t \in T, s \in S \qquad (3.29)$$

Each station has either unmet pick-up demand or unmet drop-off demand. It cannot have both (Constraints (3.30)).

$$bpp_s^t + bdp_s^t \leq 1 \qquad \forall t \in T, s \in S \qquad (3.30)$$

Unmet demand can only be greater than zero if the pick-up and drop-off penalty indicator are not zero. We have assumed that in each interval, the number of pick-ups or drop-offs cannot exceed 10 times the capacity of the stations, so M is 10. If the unmet

pick-up or drop-off demand is zero, then the corresponding binary indicator of that variable is also zero (Constraints (3.31) to (3.34)).

$$pp_s^t \leq M * R_s * bpp_s^t \qquad \forall\, t \in T, s \in S \qquad (3.31)$$

$$dp_s^t \leq M * R_s * bdp_s^t \qquad \forall\, t \in T, s \in S \qquad (3.32)$$

$$bpp_s^t \leq pp_s^t \qquad \forall\, t \in T, s \in S \qquad (3.33)$$

$$bdp_s^t \leq dp_s^t \qquad \forall\, t \in T, s \in S \qquad (3.34)$$

Constraints (3.35) and (3.36) relate the actual pick-ups (ap_s^t) and drop-offs (ad_s^t) to the number of pick-ups (P_s^t) and drop-offs (D_s^t), and to the pick-up (pp_s^t) and drop-off penalties (dp_s^t).

$$ap_s^t = P_s^t - pp_s^t \qquad \forall\, t \in T, s \in S \qquad (3.35)$$

$$ad_s^t = D_s^t - dp_s^t \qquad \forall\, t \in T, s \in S \qquad (3.36)$$

Constraints (3.37) and (3.38) make sure the unmet demand does not exceed the pick-up and drop-off demand deficiency. Similar to constraints (3.31) and (3.32), we have assumed that in each interval, the number of pick-ups or drop-offs cannot exceed 10 times the capacity of the stations (i.e., M is 10.)

$$pp_s^t \leq -(a_s^t - P_s^t + D_s^t) + M * R_s * (1 - bpp_s^t) \qquad \forall\, t \in T, s \in S \qquad (3.37)$$

$$dp_s^t \leq -(R_s - a_s^t + P_s^t - D_s^t) + M * R_s * (1 - bdp_s^t) \qquad \forall t \in T, s \in S \qquad (3.38)$$

Constraints (3.39) are subtour elimination constraints.

$$y_{sk}^t - y_{s'k}^t + (|\mathrm{N}| - 1) * x_{s'sk}^t \leq (|\mathrm{N}| - 2) \qquad \forall t \in T, s, s' \in S \qquad (3.39)$$

Finally, all variables are ensured to belong to their domains through variable domain constraints (i.e., all binary variables $\in \{0,1\}$, all continuous variables ≥ 0, and all integer variable ≥ 0 & integer.)

3.3 *Simulation Module*

To find the available bikes at the start of each interval after repositioning (except the initial availability (SS_k) which is an input to the problem) as well as the number of unmet demands during each period, the bikeshare system is simulated using a discrete event simulation model. Discrete event simulation operates the system as a sequence of events in time. The possible events modeled here are bike pick-up by users, bike drop-off by users, pick-ups by trucks, and drop-off by trucks.

Simulating the system also allows us to evaluate the assumptions of the proposed optimization module, especially the less conservative ones, and evaluate the overall performance of adding a repositioning plan to the system. The arrival time of customers to the stations for pick-up and drop-off is based on the actual trip data of the bikeshare system.

Chapter 4: Data

4.1 *Bikeshare System*

The proposed model is implemented on the data from Capital Bikeshare. Capital Bikeshare (CaBi) is a program operated by Motivate International, Inc. and jointly owned and sponsored by the District of Columbia, Arlington County, the city of Alexandria, VA, and Montgomery County, MD. It offers different membership packages, which vary based on the membership period. CaBi was the largest bikeshare system in North America until May 2013. The system had 3200 bikes and 399 stations in the DMV area (District of Columbia, Maryland, and Virginia) in July 2016, and 3700 bikes and 463 stations in May 2017.

Based on the results of the 2016 CaBi Customer Use and Satisfaction Survey, 55% of CaBi members chose the availability of more docks or bikes at existing stations as the most needed CaBi expansion option. Also, 39% indicated that there is a need for a greater density of the stations within the existing service area. This shows the importance of having an accurate prediction model in conjunction with an optimization model for rebalancing this system.

We focus our study on the 208 stations that are located within boundaries of District of Columbia and were fully in operation during 2016 and 2017 shown in Figure 3.1-1. The black dots in Figure 4-1 correspond to the location of these stations. The Surface area of the dots is related to the capacity. The capacities of the selected stations range between 9 and 52. The mean capacity and median capacity of the selected stations are 18.4 and 18, respectively.

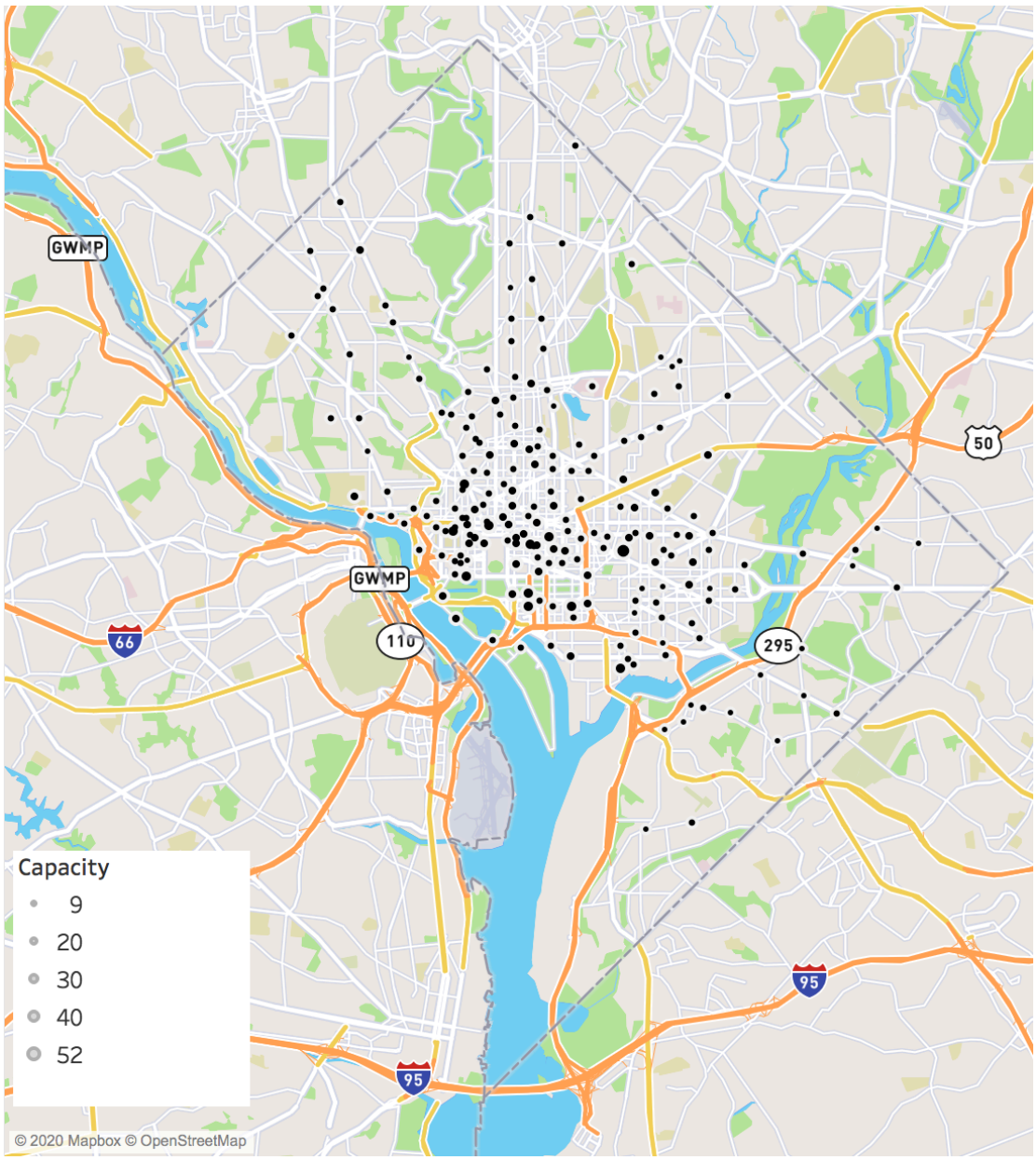


Figure 4-1 Location of the 208 CaBi Stations Used in this Study

4.2 *Trip and Membership Data*

Trip data for 2016 and 2017 is retrieved from the Capital Bikeshare website[1]. This data includes the following information regarding each trip: start date, end date, start station, end station, duration, bike number, and membership information of the user.

Membership information refers to whether the trip is taken by a member user or a casual user. Member users are those who have purchased the annual membership, which grants them unlimited trips on the shared bicycles for up to 30 minutes. Casual users are users who purchase a three-day membership, single-day membership, or a single trip. The annual membership fee is $85 (2017 CaBi pricing), three-day membership is $17, single-day membership is $8, and single trips are $2 per ride. Additional usage fees apply to both casual and members if the duration of a trip exceeds 30 minutes.

To avoid the inclusion of incorrect records in the analysis, we have done data cleaning to make sure the data includes correct trip records for the analysis. During our data cleaning process, the following trips were excluded:

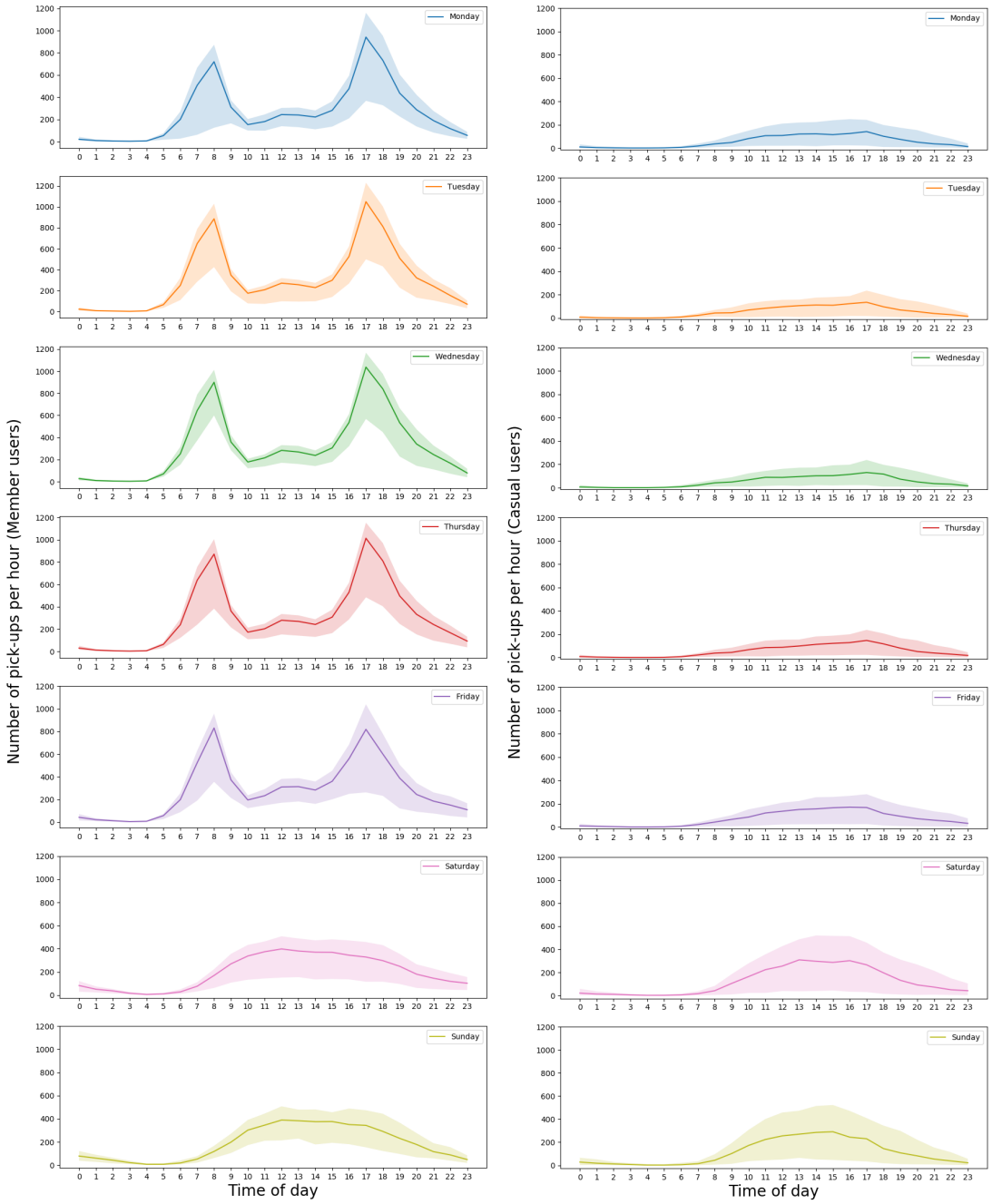
- Trips that lasted less than 60 seconds
- Trips that lasted less than 120 seconds and had the same start and end location

The number of pick-ups and drop-offs for each station were then aggregated into 30-minute or 60-minute intervals. Figure 4-2 shows the Median, 10th percentile, and 90th percentile error bands of the number of pick-ups per hour for the studied CaBi stations during 2016 and 2017.

[1] https://www.capitalbikeshare.com/

Figure 4-2(a) shows the usage of CaBi members, and Figure 4-2(b) shows the usage of CaBi casual users. We can observe a different pattern between members and casual users as well as a different pattern between weekdays and weekends. Member users' travel patterns are similar to commuting patterns during weekdays, whereas casual users' travel patterns are close to recreational travel patterns (Buck et al., 2013; Faghih-Imani et al., 2016; Zamir et al., 2019).

By looking at the pick-up and drop-off profiles of members and casual users, we can qualitatively observe that membership type can potentially contain valuable information that can assist prediction. Also, we can see that the number of trips taken by members is significantly higher than the number of trips taken by casual users during the weekdays. The median trends are more similar during weekends, but the variance of casual users' usage is slightly larger.



a) Members b) Casual users

Figure 4-2 Median, 10th Percentile, and 90th Percentile Error Bands for Number of Pick-ups Per Hour for CaBi Stations (208 selected stations) During 2016 and 2017 for a) Members and b) Casual Users

Figure 4-3 and Figure 4-4 illustrate the average number of pick-ups per hour and the average number of drop-offs per hour for the stations.

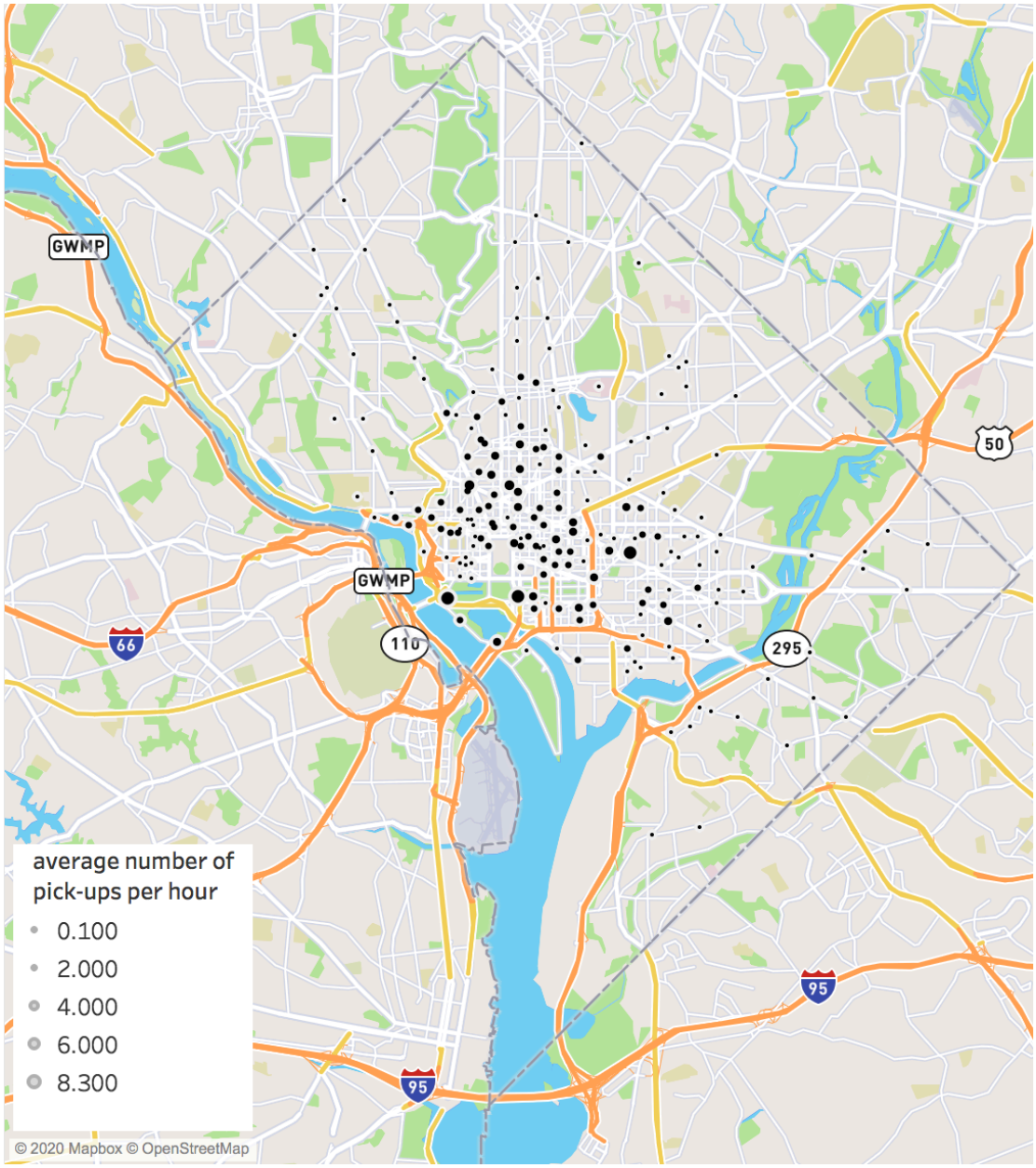


Figure 4-3 Average Number of Pick-ups Per Hour

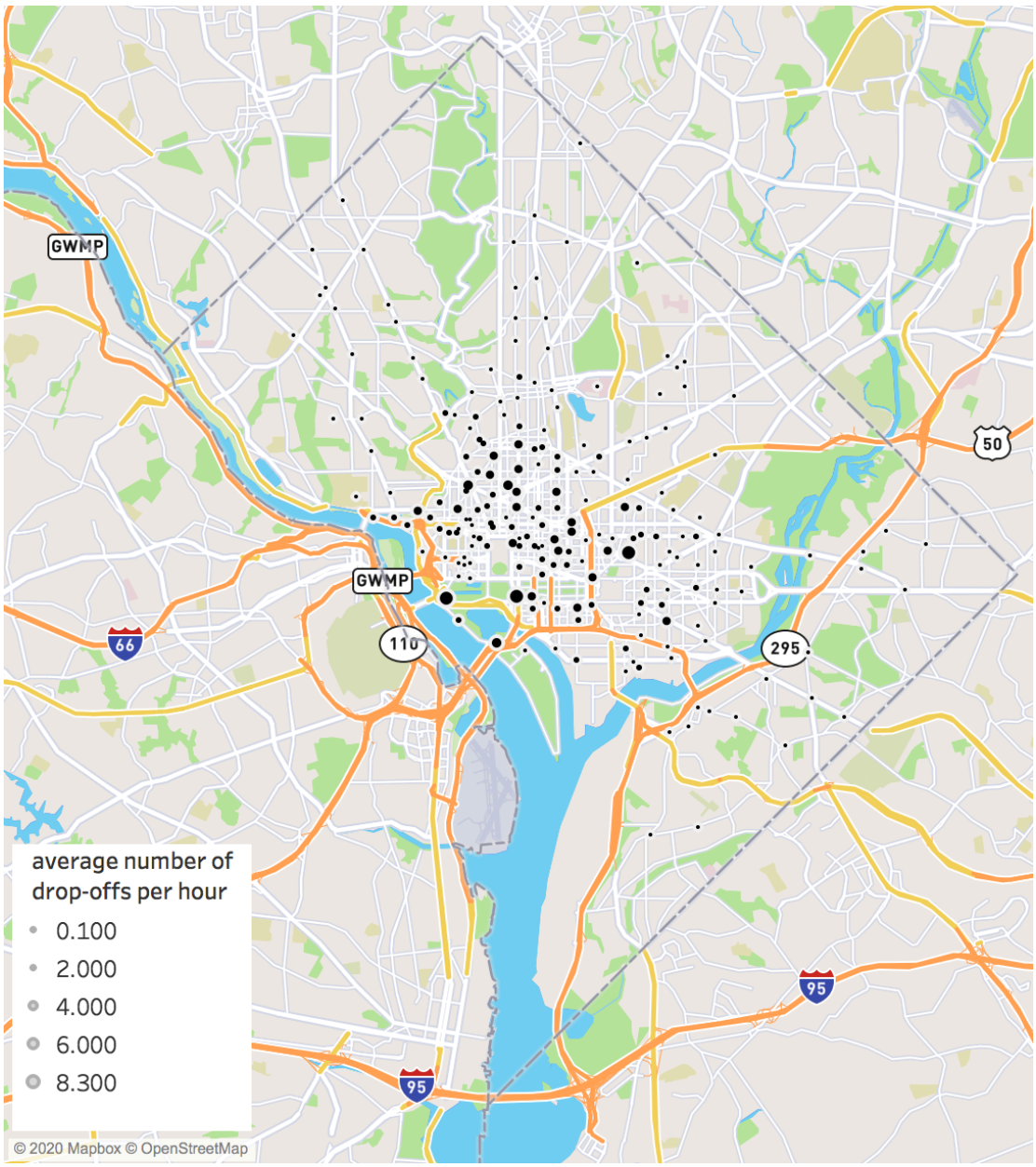


Figure 4-4 Average Number of Drop-offs Per Hour

Table 4-1 contains the summary statistics for the stations under study, and the summary statistics of the trip data.

Table 4-1 Summary Statistics for Stations

Measurement	Mean	Std.	Min	Max
Capacity	18.43[+]	6.52[+]	9	52
Average number of pick-ups per hour	1.55[+]	1.23[+]	0.02[+]	7.88[+]
Average number of drop-offs per hour	1.57[+]	1.33[+]	0.02[+]	8.28[+]
Average number of pick-ups per hour (Members)	1.19[+]	0.91[+]	0.02[+]	6.90[+]
Average number of drop-offs per hour (Members)	1.20[+]	0.98[+]	0.01[+]	7.11[+]
Average number of pick-ups per hour (Casual users)	0.36[+]	0.65[+]	0.00[+]	5.67[+]
Average number of drop-offs per hour (Casual users)	0.37[+]	0.67[+]	0.00[+]	5.68[+]
Total number of pick-ups (2016)	13,403[*]	10,665[*]	152	68,256
Total number of pick-ups (2017)	13,762[*]	11,059[*]	206	69,978
Total number of drop-offs (2016)	13,573[*]	11,439[*]	130	71,755
Total number of drop-offs (2017)	13,931[*]	11,877[*]	165	73,485

[+] Rounded off to two decimal digits
[*] Rounded off to the nearest integer

4.3 *Status of Stations*

The status of a station refers to historical data, which reports the start time and end time of the station's outages. Outage refers to being completely empty or full. Status data, which records instances that CaBi was either full or empty, is retrieved from the CaBi tracker website[2]. This website is linked to the live feed of CaBi, which gives

[2] http://cabitracker.com/

information about the number of docks and bikes available at each station. The historical dataset is available for a few of bikeshare systems including CaBi (District of Columbia, USA), Hubway (Chicago, USA), and Melbourne (Melbourne, Australia).

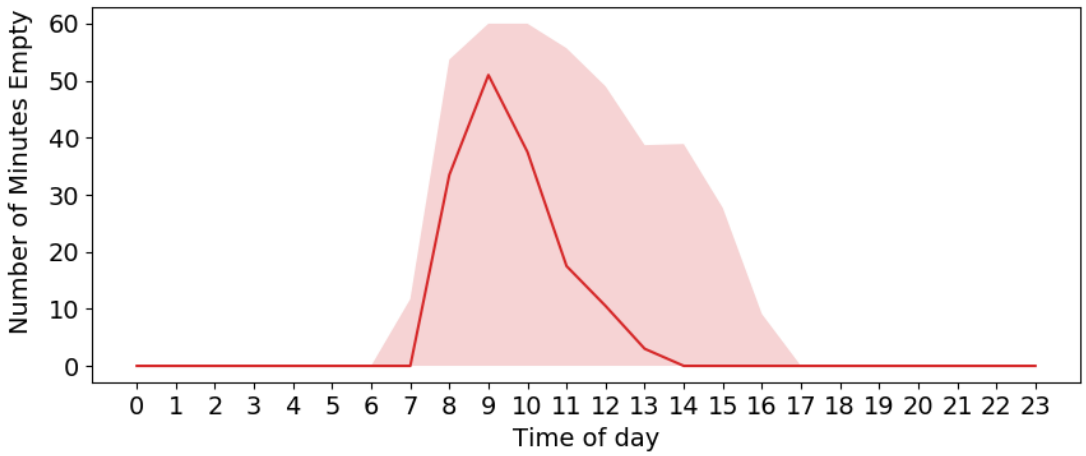
The status of station data includes station name, status, start, end, and duration. Status can have two values empty or full. Empty means that no bikes are available for pick-up, and full means that no docks are available for drop-off. The start column gives information about the start time that a specific station's status is changed to full or empty, and the end column gives information about the end time of that status (after end time, the station has bikes and docks ready for pick up and drop-off). The precision of start time and end time is 1 minute. We clean the data by removing duplicate and overlapping records. We also process the data by calculating the duration of having full or empty stations in each 30-minute interval. After processing there are two columns for each station, one for recording full instances and one for empty instances, which indicates duration of being full or empty during each 30-minute interval. Table 4-2 summarizes the total duration of the outage (empty/ full) for 2016 and 2017 aggregated over all 208 stations.

Table 4-2 Summary Statistics of Outage for 2016-2017

	2016				**2017**			
	Mean	**Std.**	**Min**	**Max**	**Mean**	**Std.**	**Min**	**Max**
Full (Minutes)	16,922*	14,522*	0	58,012	19,465*	18,396*	0	92,774
Empty (Minutes)	33,286*	22,765*	124	93,580	42,767*	28,727*	37	125,975

*Round off to the nearest integer

Figure 4-5 illustrates an example of the status data for a station of CaBi (8th & O St NW). Figure 4-5(a) shows the median, 10th percentile, and 90th percentile for the number of minutes that the station was empty throughout the day during 2016 and 2017 on Thursdays, and Figure 4-5(b) shows the average number for pick-ups during the same period. As presented, this station experiences outage for a significant amount of time between 8 AM and 12 PM. As a result, the number of pick-ups during this period is highly unreliable (i.e., is not fully reflective of the number of actual demand) and could be misleading for training a prediction model (O'Mahony and Shmoys, 2015). To include temporal outage information, we extract outage data for each interval. By including this source of information, we believe that the model will learn to predict lower usage in the presence of outage. It should be noted that for predicting the future demands to be used in rebalancing operations (implementation mode), outages should be set to zero. In other words, for applying the model, we want to predict the demand of stations so that we do not experience outages.



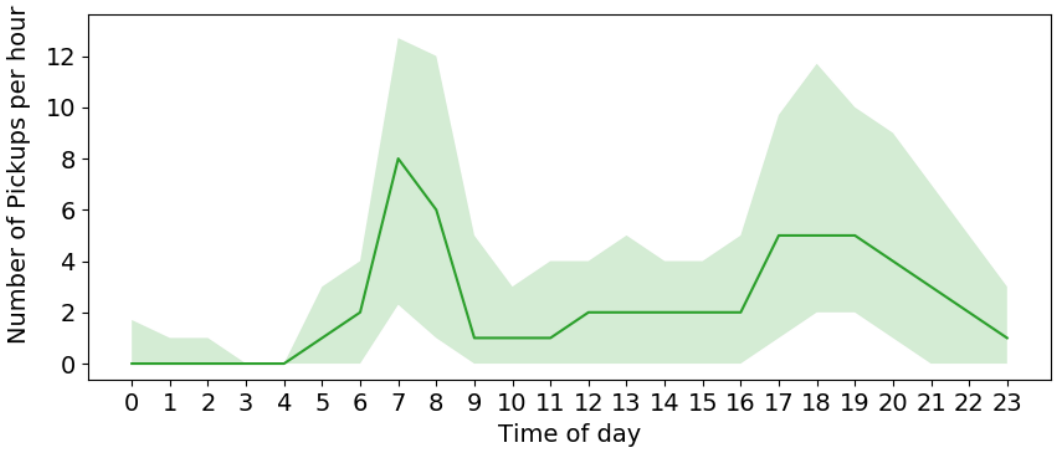


Figure 4-5 a) Median, 10th Percentile, and 90th Percentile Error Bands for a) Number of Minutes that Station (8th & O St NW) is Empty (Thursdays) b) Number of Pick-ups Per Hour (Thursdays) at the Same Station

4.4 *Weather Data*

Weather data, which includes hourly precipitation data in inch, hourly dry bulb temperature data in Fahrenheit, hourly relative humidity in percent, and hourly wind speed in mile per hour, is retrieved from NOAA's (National Oceanic and Atmospheric Administration) National Centers for Environmental Information (NCEI). The records are for Ronald Regan Washington National Airport, which is located in the proximity of the District of Columbia. We only used the FM-15 report type as it includes data that is recorded on regular 1-hour intervals and does not cause inconsistencies to our models' input data. These records are used to estimate the weather for each 30-minute interval. For temperature, humidity, and wind speed, we use interpolation and average the values for the 30-minute intervals before and after. We have assumed that precipitation is uniformly distributed, and we split it for each 30-minute interval. Although this assumption might not be true, it is reasonable, as studies in the literature

have confirmed that the demand for a given time-interval of bikeshare systems is correlated with weather data from the time intervals before and after it (Reiss and Bogenberger, 2015). Table 4-3 shows the summary statistics of weather data for 2016 and 2017. Based on this table, we observe that there isn't much variation between weather measurements in 2016 and 2017.

Table 4-3 Summary Statistics of Weather Data for 2016-2017

	2016				**2017**			
	Mean	**Std.**	**Min**	**Max**	**Mean**	**Std.**	**Min**	**Max**
Temperature (Fahrenheit)	59.9	18.1	13	100	60.4	17.1	15	97
Precipitation (Inch per half an hour)	0.001	0.013	0.000	0.390	0.002	0.015	0.000	0.370
Wind speed (Miles per hour)	8.7	4.9	0.0	39	8.7	4.9	0.0	34
Relative humidity (Percent)	64.3	19.2	15	100	65.2	19.7	13	100

Chapter 5: Prediction Module

5.1 *Overview*

As indicated in section 2.2, different machine learning and statistical techniques exist for predicting time series. Appendix A includes a comparison of some of these models, namely, dynamic regression, random forest, and deep neural networks (DNNs) in our case study. Among these models, the DNN achieved the lowest mean squared error. As a result, this model is chosen for further consideration and detailed parameter tuning.

5.2 *Hyperparameter Tuning and Model Selection for DNN*

A Random grid search in the hyperparameter space is deployed for finding a good combination of hyperparameter values. Hyperparameter tuning is performed on the following variables:

Number of hidden layers: number of layers in the graph in addition to input and output layer

Number of neurons: number of processing units in each layer

Dropout rate: the rate of eliminating some number of neurons of the hidden layers randomly during training. This method is used to reduce over-fitting and improve generalization error.

Fixed values are used for other parameters of the model, including the number of epochs, which refers to the number of times all of the training vectors are used once to update the weights and the parameters of the Adam optimizer which is the optimization algorithm used for training the neural network model.

Given that every station will have its own model for the number of pick-ups and drop-offs, a pipeline is developed for our search, which greatly speeds up the hyperparameter tuning step. The stations are divided into three groups: low usage, medium usage, and high usage. These groups are defined as follows:

- Low usage: stations that their average pick-up and drop-off (both) per hour is less than 1.
- Medium usage: stations that their average pick-up and drop-off (both) is more than 1 and less than 2.
- High usage: stations their average pick-up and drop-off (both) are more than 2 per hour.

The thresholds mentioned above are selected to result in balanced clusters. Based on the mentioned criteria, 72 stations are identified as low usage, 68 stations are identified as medium usage, and 68 stations are identified as high usage. 20 percent of stations are randomly selected from each group, and a random search for hyperparameter tuning is performed on them. For each station, 10 random combinations of hyperparameters are chosen. The best combination for each station is then selected based on the values of root mean squared error of the validation dataset. Each station is then trained based on the best combination of other stations in its group, and the hyperparameter that has the best average performance across all the stations from that group is picked as the final hyperparameter. Figure 5-1 summarizes the hyperparameter tuning workflow.

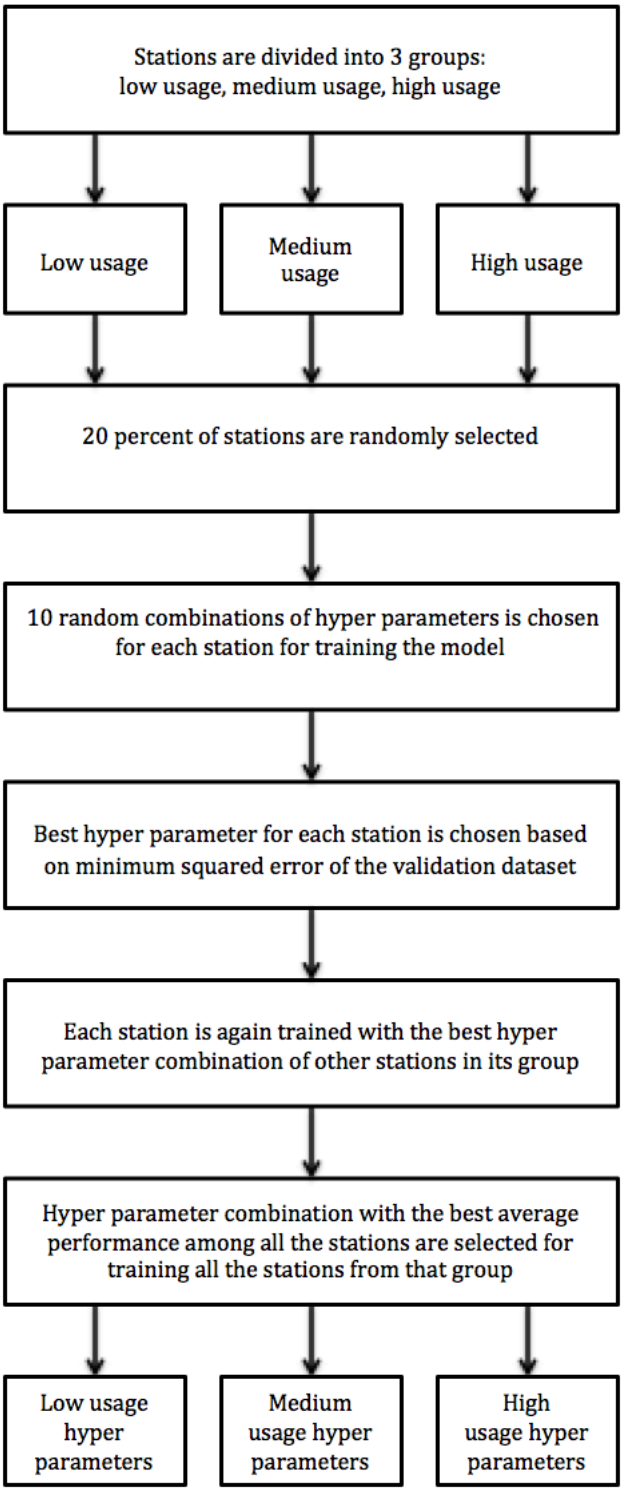


Figure 5-1 Hyperparameter Tuning Workflow

More details and visual summaries regarding hyperparameter tuning can be found in Appendix B. Table 5-1 summarizes the hyperparameters and parameters resulting from this random search method for each usage group.

Table 5-1 Hyperparameters and Parameters Value for Each Station Group

		Usage group	
Parameters	**Low usage**	**Medium usage**	**High usage**
Number of layers	2	3	4
Number of neurons (1st hidden layer)	25	25	25
Number of neurons (2nd hidden layer)	50	200	25
Number of neurons (3rd hidden layer)	-	250	250
Number of neurons (4th hidden layer)	-	-	250
Drop rate	0.4	0.4	0.6
Number of past observations	12	2	6

5.3 *Dynamic Prediction and Weight Re-estimation in the DNN*

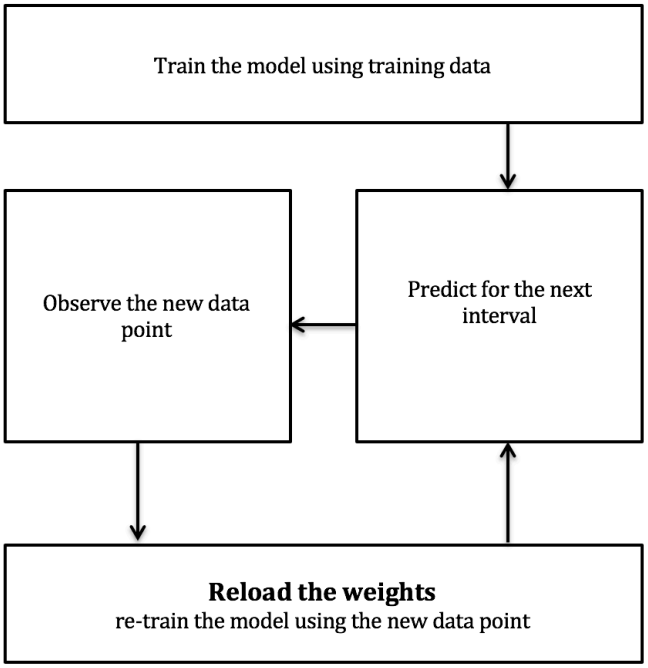
Two methods are tested for re-estimating the weights in the neural network to take advantage of the streaming data and improving the predictions.

5.3.1 Training with Model Reloading

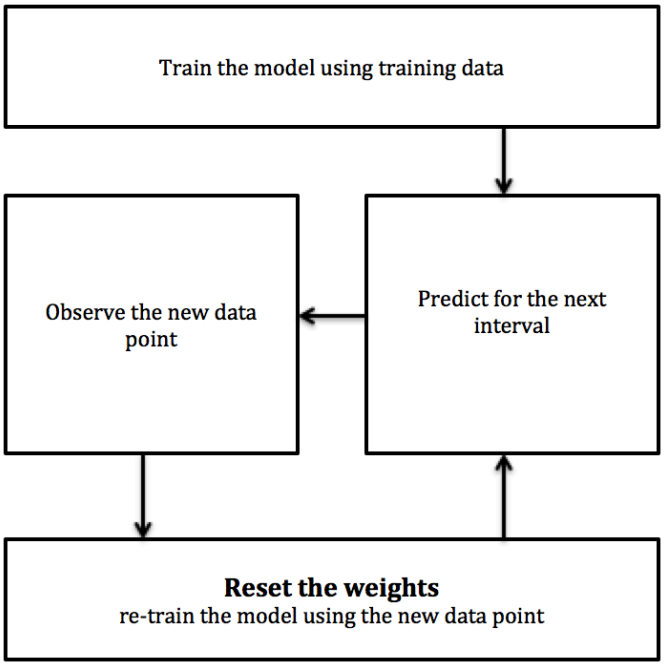
Figure 5-2(a) summarizes the dynamic setting workflow for the first method. In this method, the neural network is trained on the training dataset for a number of epochs, and weights of the model are then stored for each station. With every new data point (i.e., aggregated number of pick-ups within 30 minutes), the weights are updated/fine-tuned by running another epoch on the updated, augmented dataset. The updated model with new weights is then used for predicting the new interval. This approach may have the disadvantage of overfitting due to the excessive number of forward and backward passes (number of epochs) on the data.

5.3.2 Training with Model Resetting

Figure 5-2(b) summarizes the dynamic setting workflow for the second method. In this method, with every new data point, weights are reset to the model that has only used the initial training dataset. The model is then trained for a fixed number of epochs on the new updated and augmented dataset. This method limits the number of passes on the dataset and could overcome the overfitting problem that exists in the first approach. However, since this model needs to reset the weights and could potentially require more than one additional epochs, it could need more time for fine-tuning and could be slower compared to the first approach.



a) Training with model reloading



b) Training with model resetting

Figure 5-2 Dynamic Setting Workflow for a) Training with Model Reloading b) Training with Model Resetting

5.4 *Multi-step Forecasting*

Due to the nature of the optimization module (multi-step model with a rolling horizon) described in detail in Chapter 3 and to capture the fluctuation in the number of pick-ups and drop-offs of the stations, we need to predict the demand periodically. As a result, the prediction module needs to predict not only the usage for the next interval but also other future intervals. Here, a 2-hour horizon is used. In sections 5.4.1 and 5.4.2, two different duration intervals are used for predicting this 2-hour horizon. In section 5.4.1, 30-minute intervals are used, whereas section 5.5.2 uses 1-hour intervals. Thus, for the 30-minute model, we need to predict the system for the next 4 time intervals, whereas for the 1-hour model, we need to predict the system for only 2 time intervals. If the intervals are short, the rebalancing operation is more aligned with the assumptions of the optimization module (see 3.1.1). However, this could potentially result in lower accuracy for the predictions of the model. Sections 5.4.1 and 5.4.2 compare the prediction accuracy of these two models. Prediction module results

For all of the following sections except section 5.4.3 (Effect of dynamic set-up), 2016 Capital Bikeshare trip data is used for training the models, and 2017 Capital Bikeshare trip data is used to test and compare them in terms of performance. Although feedforward neural networks do not explicitly require sequential time-series data, we decided to follow the standard method of time series validation that is training the model sequentially and without any interruptions. Using 1 year of data for training and 1 year of data for testing allows us to account for seasonality, which makes using an un-shuffled training dataset more reasonable and closer to the real world.

Prediction accuracy of models are evaluated in term of root mean squared error (RMSE).

$$RMSE = \sqrt{\frac{\sum_{i=1}^{i=n} (P_i - O_i)^2}{n}} \tag{5.1}$$

where n is the length of time series (time is indexed by i.) P_i is the prediction for the number of pick-ups (or the number of drop-offs) at time index i. O_i is the observed number of pick-ups (or number of drop-offs) at time index i.

5.4.1 Effect of Adding Status Information and Membership Information

Results for this section are based on multi-step forecasts (4-steps) without re-estimation meaning the model is estimated based on a single set of training data, and the estimated model is used to forecast the number of pick-ups and drop-offs for the next four 30 minute intervals for the test dataset. To test the value of adding membership and status information, we have trained separate models for the number of pick-ups/drop-offs by members versus casual users and compared that with a base model that makes predictions by ignoring this information.

Figure 5-3 and Figure 5-4 show the average root mean squared error along with the variance (whisks) for each model and across all of the studied stations for the number of pick-ups and drop-offs, respectively. The x-axis shows the time step for which we are reporting. The base model has the highest average root mean squared error. The base model uses weather data (precipitation, temperature, relative humidity, and wind speed), time of day, day of the week, and monthly indicators as inputs. Time of day is

incorporated into the model by dividing the day into 48 30-minute intervals. We use one-hot encoding, which is a vector representation of categorical variables using 0s and 1s, to include these variables into the model. We also include the number of pick-ups or drop-offs from the previous intervals (similar to the autoregressive part of dynamic regression models) as input features of the model.

The model with only status information has the second highest root mean squared error. Producing different models for different memberships come next, and the models with all of the information have the lowest root mean squared error. When interpreting the results, we should keep in mind that for the models/cases in which we have two separate models for member and casual users, the overall root mean squared error is between the root mean squared error of members and casual users. Depending on the usage split between casual and members that exists at each station, this number could be close to casual users or member users. Root mean squared error is relatively constant (only slightly increases for later time steps) between the time steps. Including status improves prediction marginally, and results indicate including status is more beneficial for the near future time-steps.

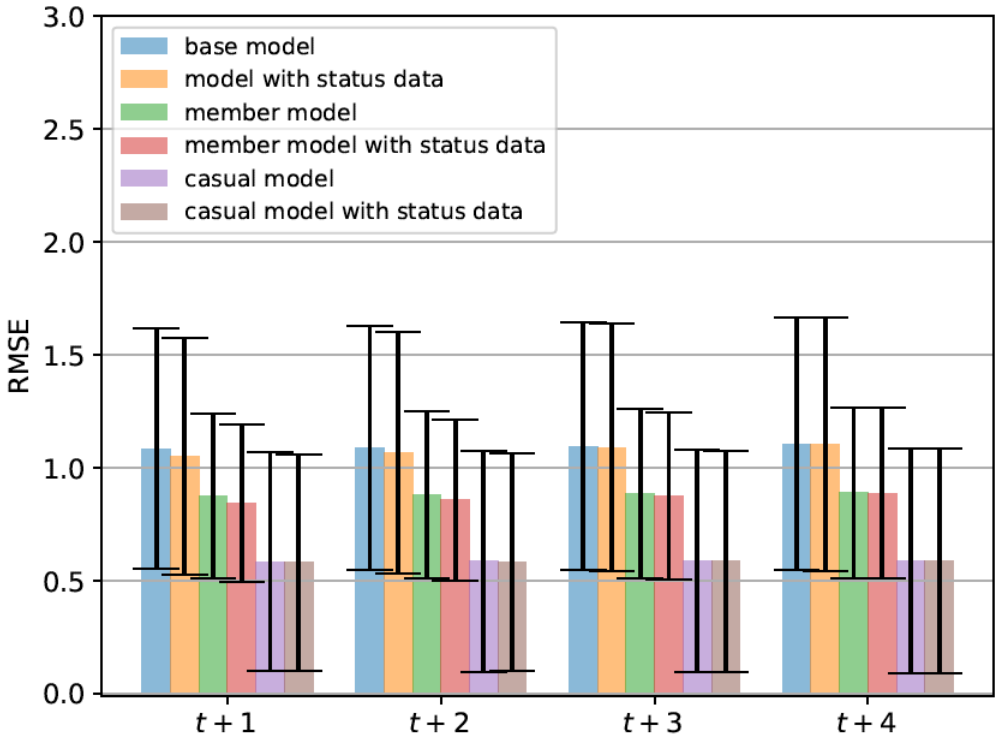


Figure 5-3 Average Performance (Root Mean Squared Error) of All Tested Models on Future Time Interval Predictions (Half an Hour) Across All of the Studied Stations for Pick-ups

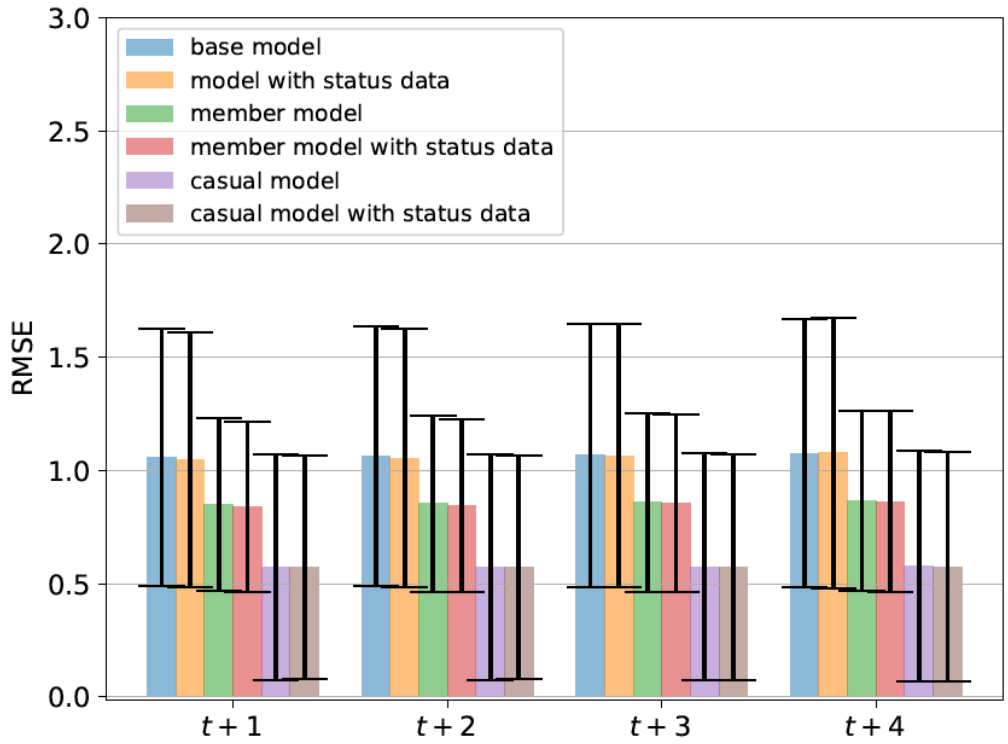


Figure 5-4 Average Performance (Root Mean Squared Error) of All Tested Models on Future Time Interval Predictions (Half an Hour) Across All of the Studied Stations for Drop-offs

Figure 5-5 illustrates the kernel density estimate of the average root mean squared error across all studied stations. The density of the distribution for the model with membership information and status information is concentrated on the left of distribution for the base model, which indicates the additional information has reduced the overall root mean squared error for the predictions.

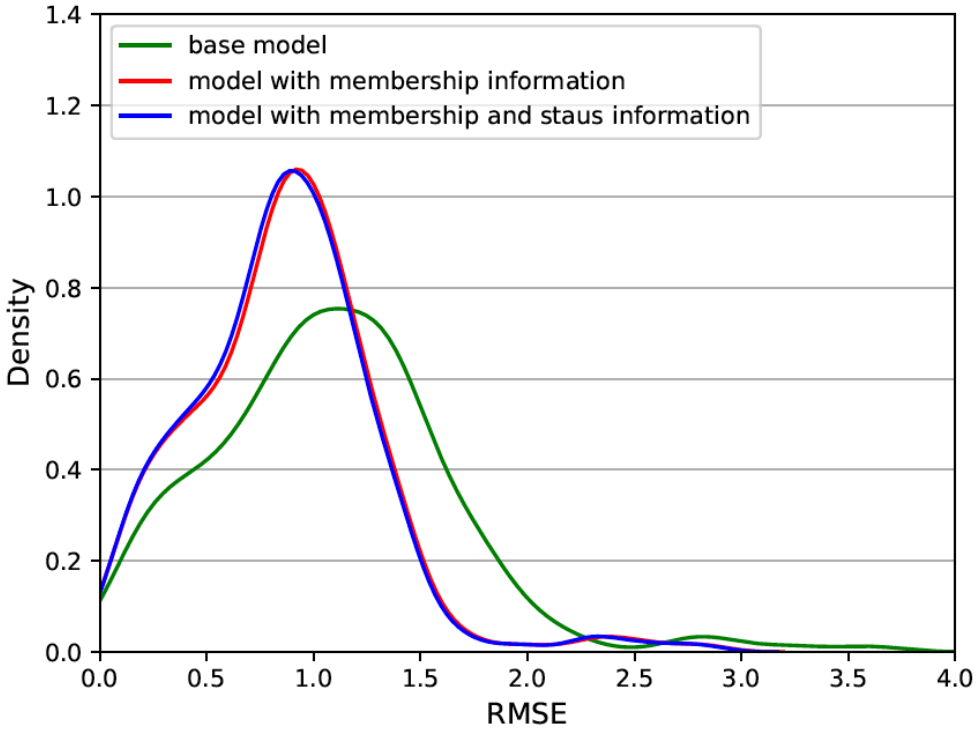


Figure 5-5 Kernel Density Estimate of Average Root Mean Squared Error (Half an Hour) Across All of the Studied Stations (for Pick-ups and Drop-offs)

Figure 5-6 shows the improvement of root mean squared error (in terms of percentage change) by adding membership and status information. This plot accounts for the split usage that exists between the number of pick-ups or drop-offs by the member and casual users of each station and gives us a concrete estimation of improvements resulting from the proposed model. As the figure suggests, a 30% improvement has taken place for some of the stations. The median improvement is around 19%.

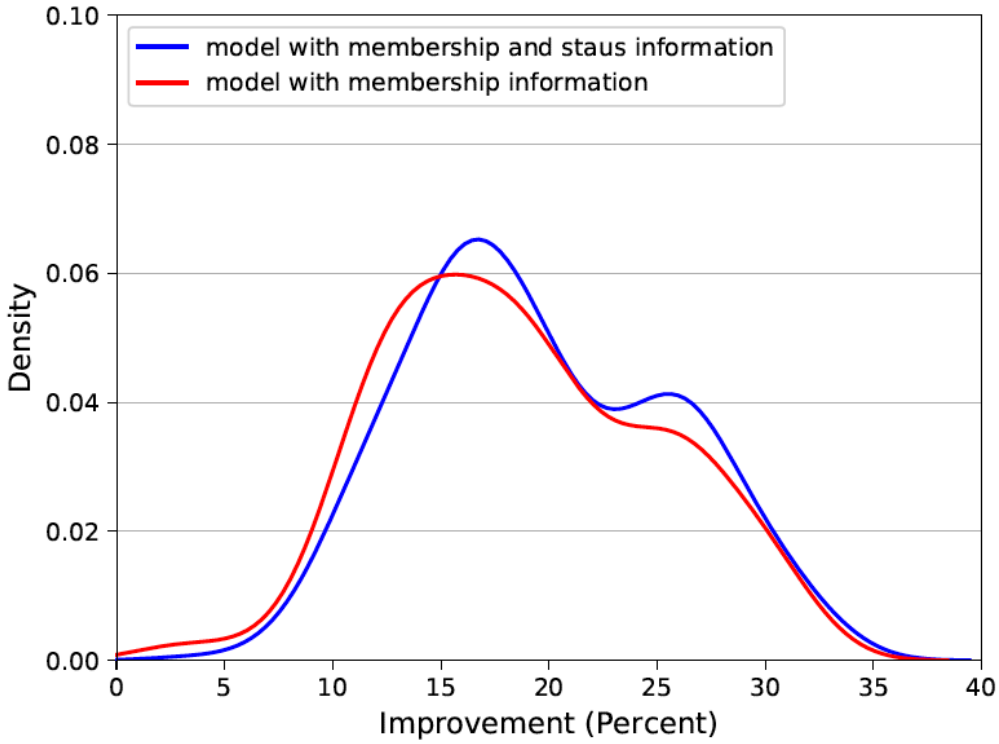


Figure 5-6 Kernel Density Estimate of Percent of Improvement in Root Mean Squared Error Among All of the Studied Stations and Time Steps (Half an Hour) When Compared to the Base Model (for Pick-ups and Drop-offs)

For a more rigorous comparison, we compare whether the improvement seen in Figure 5-6 is statistically significant. For this, we used the Wilcoxon signed-rank test. The p-value suggests that we could reject the null hypothesis at a confidence level of 5%, concluding that model with membership and status information has a lower root mean squared error compared to the base model.

5.4.2 Effect of Aggregation

Results for this section are based on multi-step forecasts (2-steps) without re-estimation, and the estimated model is used to forecast the number of pick-ups and drop-offs for the next two 1-hour intervals for the test dataset. Aggregation could be used in time series forecasting to increase the signal/noise ratio.

Figure 5-7 and Figure 5-8 show the average root mean squared error along with the variance (whisks) for each model and across all of the studied stations for the number of pick-ups and drop-offs, respectively. Similar to section 5.4.1 results, the base model has the highest expected root mean squared error, and models with both status and membership information have the lowest root mean squared errors.

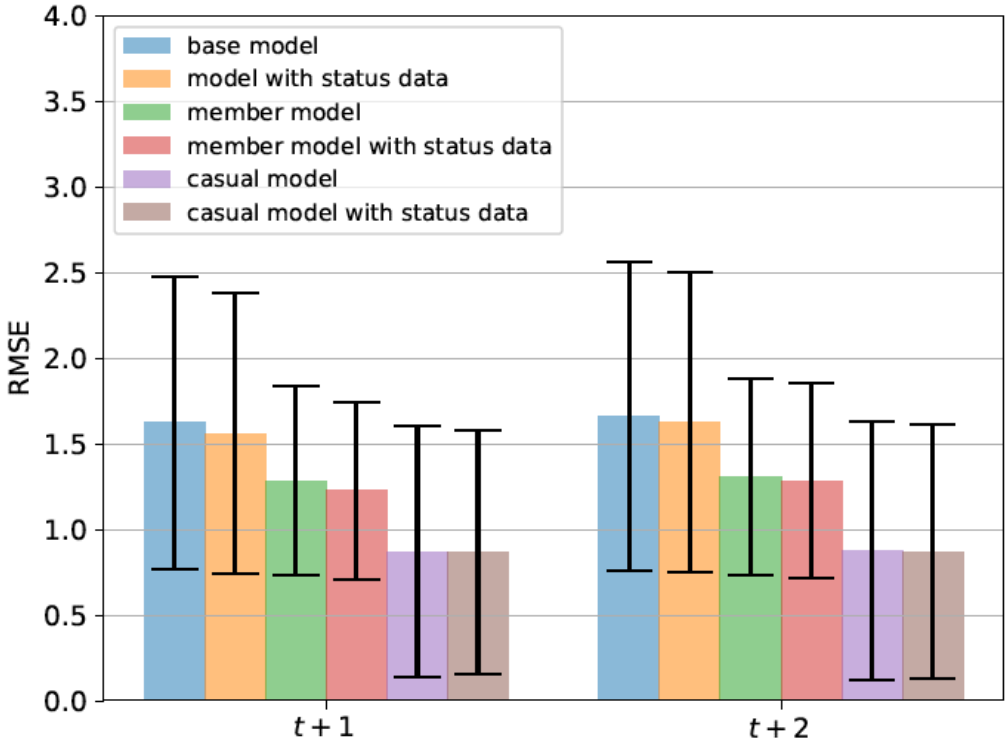


Figure 5-7 Average Performance (Root Mean Squared Error) of All Tested Models on Future Time Interval Predictions (Hourly) Across All of the Studied Stations for Pick-ups

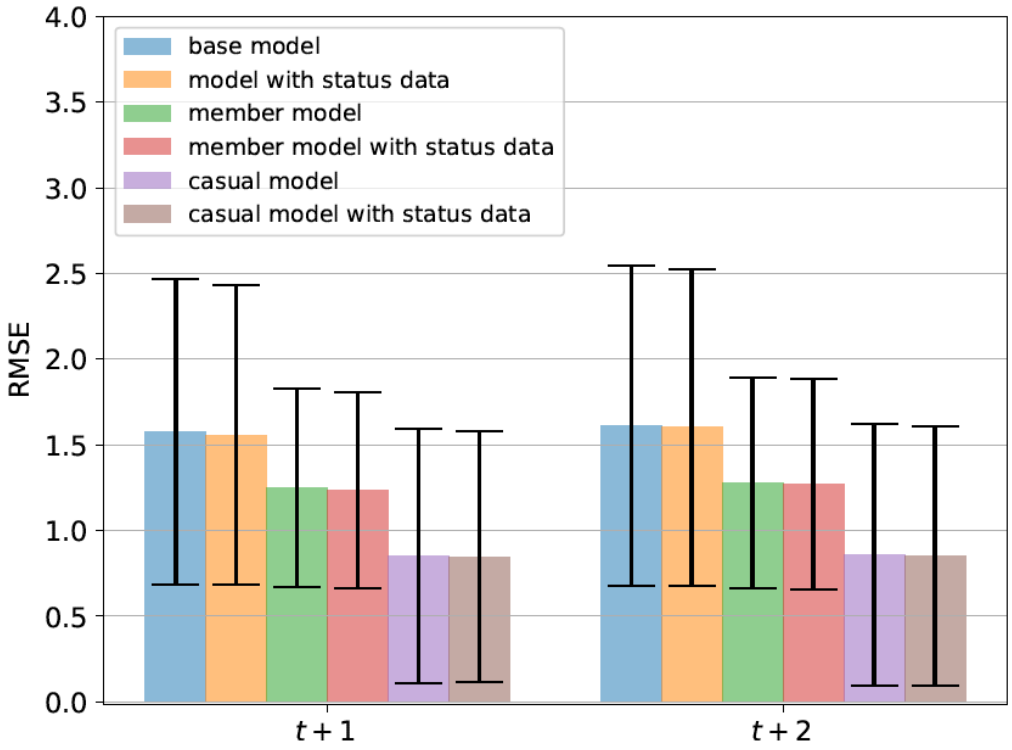


Figure 5-8 Average Performance (Root Mean Squared Error) of All Tested Models on Future Time Interval Predictions (Hourly) Across All of the Studied Stations for Drop-offs

Figure 5-9 shows the improvement to root mean squared error (in terms of percentage change) by adding membership and status information.

To see the effect of aggregation on the root mean squared error, the sum of root mean squared error for future time steps of the model with 30-minute intervals is compared with that of the hourly interval method. Figure 5-10 shows the percentage improvement in the root mean squared error by aggregating the data hourly. The median improvement is around 27%.

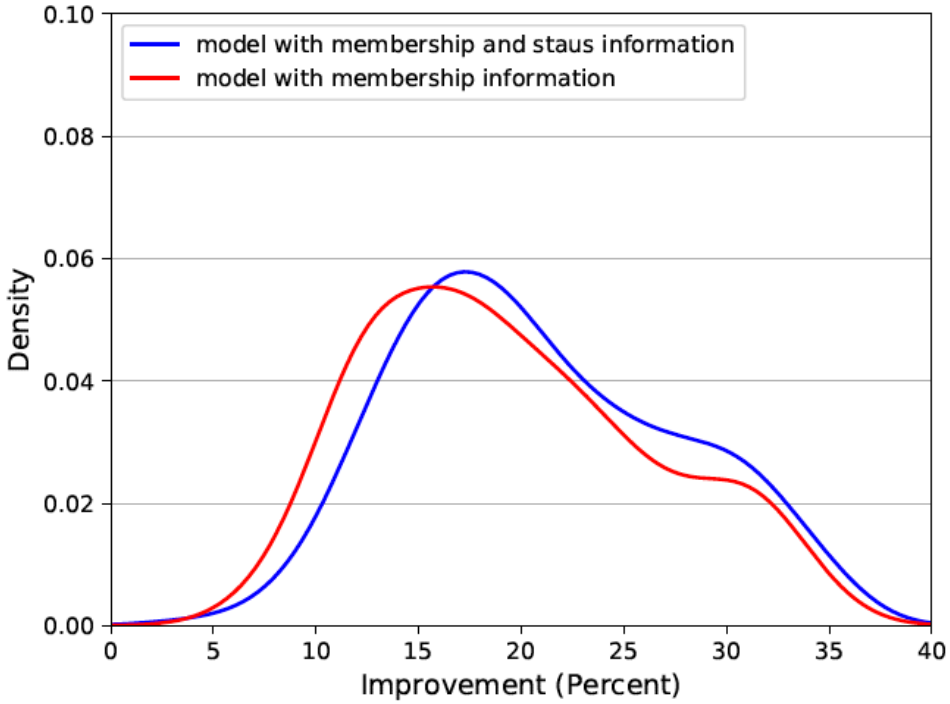


Figure 5-9 Kernel Density Estimate of Percent of Improvement in Root Mean Squared Error Among All of the Studied Stations and Time Steps (Hourly) when Compared to the Base Model (for Pick-ups and Drop-offs)

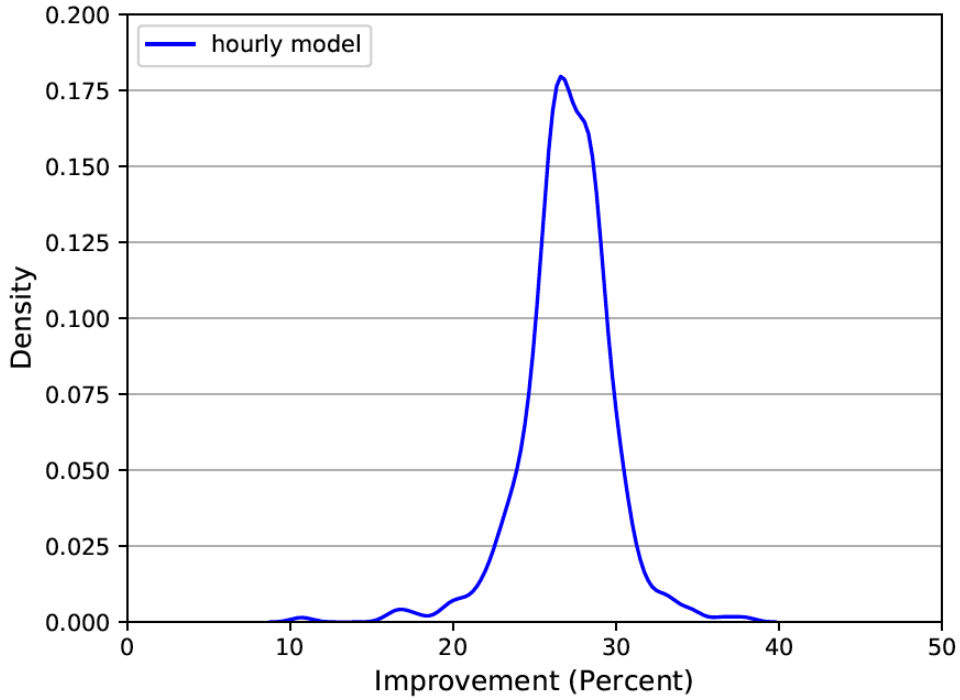


Figure 5-10 Kernel Density Estimate of Percent of Improvement in Root Mean Squared Error Among All of the Studied Stations and Time Steps (Hourly)

5.4.3 Effect of Dynamic Set-up

Figure 5-11 illustrates the percentage change to root mean squared error by using the dynamic approach in 5.3.2 on a week of the dataset. The results indicate in most cases, the dynamic model will have better predictions.

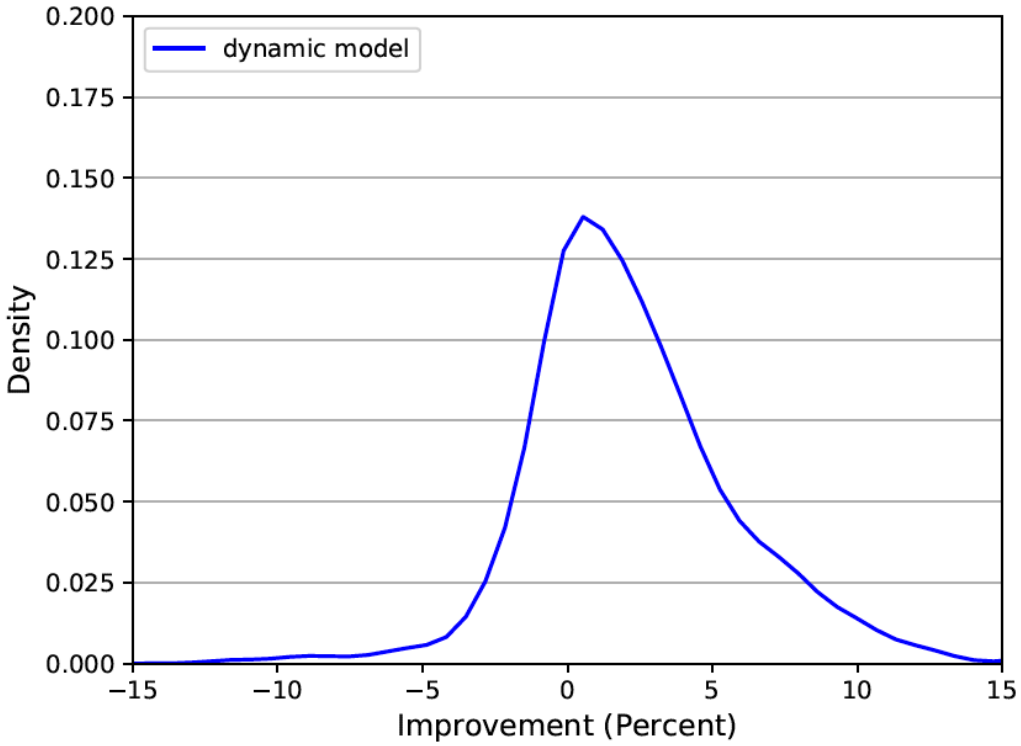


Figure 5-11 Kernel Density Estimate of Percent of Improvement in Root Mean Squared Error Among All of the Studied Stations and Time Steps (Half an Hour) for the Dynamic Case

Chapter 6: Numerical Results

6.1 *Overview*

The proposed optimization module is coded in Python using the gurobipy library[3], which is Guorbi's python interface. After running each round of the optimization module, the bikeshare system is simulated for measuring the performance of repositioning as well as calculating the initial availability at the start of the next round. The simulation module is also coded in Python to make this procedure convenient.

For this chapter and the following chapter, the numerical experiments are chosen so that number of total pick-ups and total drop-offs for the selected stations do not deviate much from each other for the studied time of day. Given the initial number of total bikes for the numerical experiments is within 40% to 60% of the capacity of the selected stations, if these two numbers deviate much, it will result in a lack of bikes or docks in the system. For example, if the number of pick-ups is much higher than the number of drop-offs towards the end of the period, the number of bikes in the system will be very low, which is not well aligned with the real-world instances as the bikeshare systems are closed systems. As a result, numerical experiments are chosen so that the total number of pick-ups for the whole period is within 0.7 to 1.3 of the total number of drop-offs.

The results of the optimization module tested on the observed pick-ups and drop-offs from 20 stations are presented here to verify the improvements resulting from the

[3] https://www.gurobi.com/documentation/

modeling and to evaluate the assumptions of the model. Figure 6-1 and Figure 6-2 show the locations of the selected stations and their associated IDs, respectively.

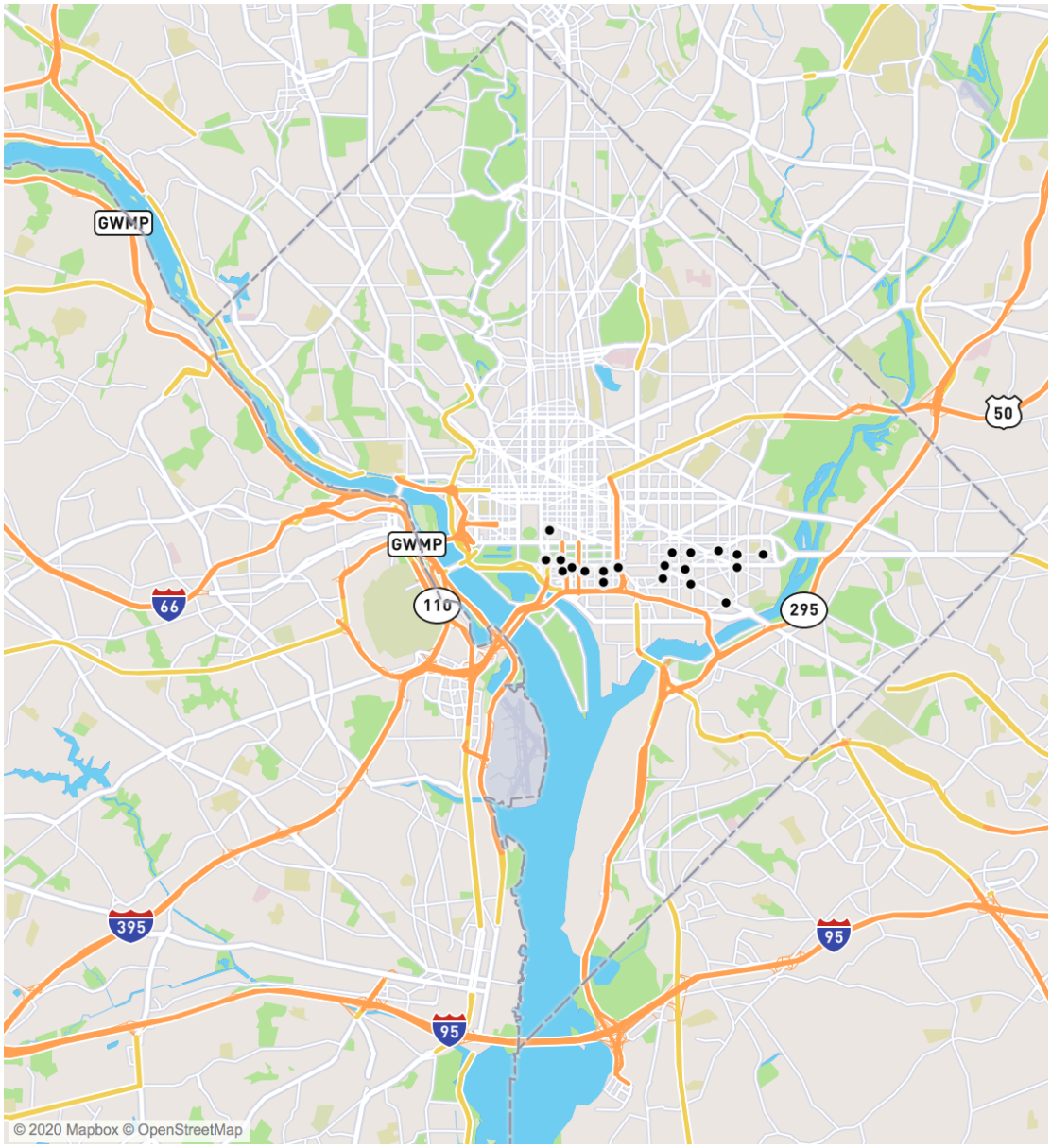


Figure 6-1 Location of Stations for the Numerical Experiments

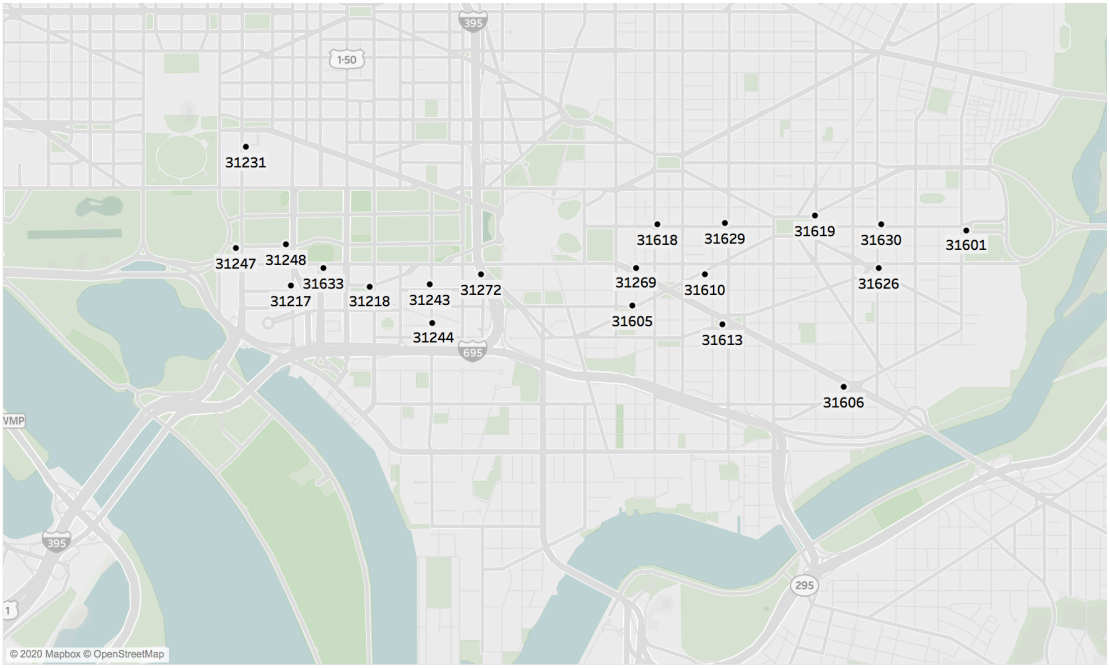


Figure 6-2 Stations' ID for the Numerical Experiments

Three periods are included in the analysis to test the validity of the assumptions during different times of the day: a weekday morning peak (6 am-10 am), a weekday afternoon peak (3 pm-7 pm), and a weekend mid-day peak (11 am-4 pm). The optimization module generates the truck's pick-ups and drop-offs and the routing schedules based on the next 2 hours. Two different interval durations are tested for the optimization module: 30 minutes (4 intervals) and 60 minutes (2 intervals).

The repositioning is done using one truck with a capacity of 20 bikes, and the starting location of the truck is terminal number 31606. For travel time, estimates from Google Maps are used. α and β are set to 0.01 in the objective function.

Two different methods for generating the initial number of bikes (initializations) at each station is used. The first method fixes the initial number of bikes uniformly to 40%, 50%, and 60% of the capacity for each station. In the second method, the initial

number of bikes are randomly generated with the constraint that the ratio of the total number of bikes over the total capacity for a station is between 45% and 55%. For the second method, random numbers are generated three times.

Table 6-1 summarizes the characteristics of the presented numerical experiments including the time of day as defined before, the ratio of the initial number of bikes in the station to the capacity of the station, and total demand, which is the sum of the number of pick-ups and drop-offs during the time of day of the case study.

Table 6-1 Characteristics of Numerical Experiments

Numerical experiment	**Time of day**	**Interval duration**	**Initialization method**	**Total demand**
Case 6-1	weekday morning peak	30 minute	random (45%-55%)	531
Case 6-2	weekday morning peak	30 minute	uniform (40%,50%,60%)	531
Case 6-3	weekday morning peak	60 minute	random (45%-55%)	531
Case 6-4	weekday morning peak	60 minute	uniform (40%,50%,60%)	531
Case 6-5	weekday afternoon peak	30 minute	random (45%-55%)	1117
Case 6-6	weekday afternoon peak	30 minute	uniform (40%,50%,60%)	1117

Case 6-7	weekday afternoon peak	60 minute	random (45%-55%)	1117
Case 6-8	weekday afternoon peak	60 minute	uniform (40%,50%,60%)	1117
Case 6-9	weekend mid-day peak	30 minute	random (45%-55%)	680
Case 6-10	weekend mid-day peak	30 minute	uniform (40%,50%,60%)	680
Case 6-11	weekend mid-day peak	60 minute	random (45%-55%)	680
Case 6-12	weekend mid-day peak	60 minute	uniform (40%,50%,60%)	680

6.2 *Optimization Module Results*

Table 6-2 stores the detailed output of the optimization module for one of the instances of the first numerical experiment (Case 6-1). This includes the order of stations visited and the number of pick-ups or drop-offs (inside the parenthesis). For example, during the first interval (6:00-6:30), the truck pickups 6 bikes from station 31606 and 2 bikes from station 31244, and drops 4 bikes at station 31218 and 4 bikes at 31231.

Table 6-2. Optimization Module Results for Case 6-1

Time	**Visited stations (Corresponding pick-ups or drop-offs)**

6:00-6:30	31606 (6)-31244 (2)-31218 (-4)-31231(-4)
6:30-7:00	31231 (8)-31247 (-8)-31633 (2)-31272 (8)-31618 (4)-31629 (5)-31606 (1)
7:00-7:30	-
7:30-8:00	31231 (1)-31247 (-1)
8:00-8:30	31247 (4)-31217 (7)-31633 (2)-31243 (7)-31605 (-9)-31269 (-2)-31613 (-4)-31606 (-5)
8:30-9:00	-
9:00-9:30	31247 (1)-31243 (7)- 31244 (-8)
9:30-10:00	31243 (9)-31218 (-9)

6.3 *Simulation Module Results*

We run the simulation without any repositioning to evaluate the improvements in the efficiency of the bikeshare system as a result of repositioning the bikes in the system.

Figure 6-3, Figure 6-4, and Figure 6-5 show the mean and variance for the number of unmet demands with and without any repositioning plan for weekday morning peak, weekday afternoon peak, and weekend mid-day peak. The means and variances are calculated based on different initializations.

Overall, random initializations result in a higher number of unmet demands compared to uniformly distributed initializations. The overall performance of the model is better during weekdays compared to weekends. This may suggest the assumptions of the model do not hold very well during weekends, and shorter intervals should be picked for the interval durations.

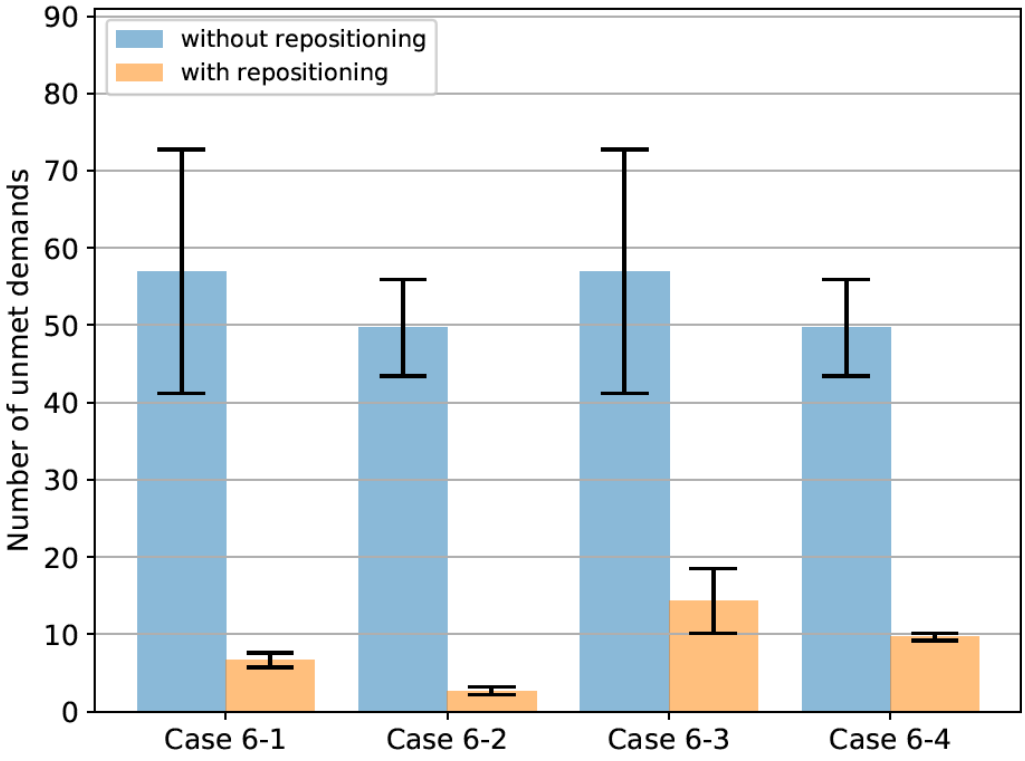


Figure 6-3 Weekday Morning Peak Numerical Experiments

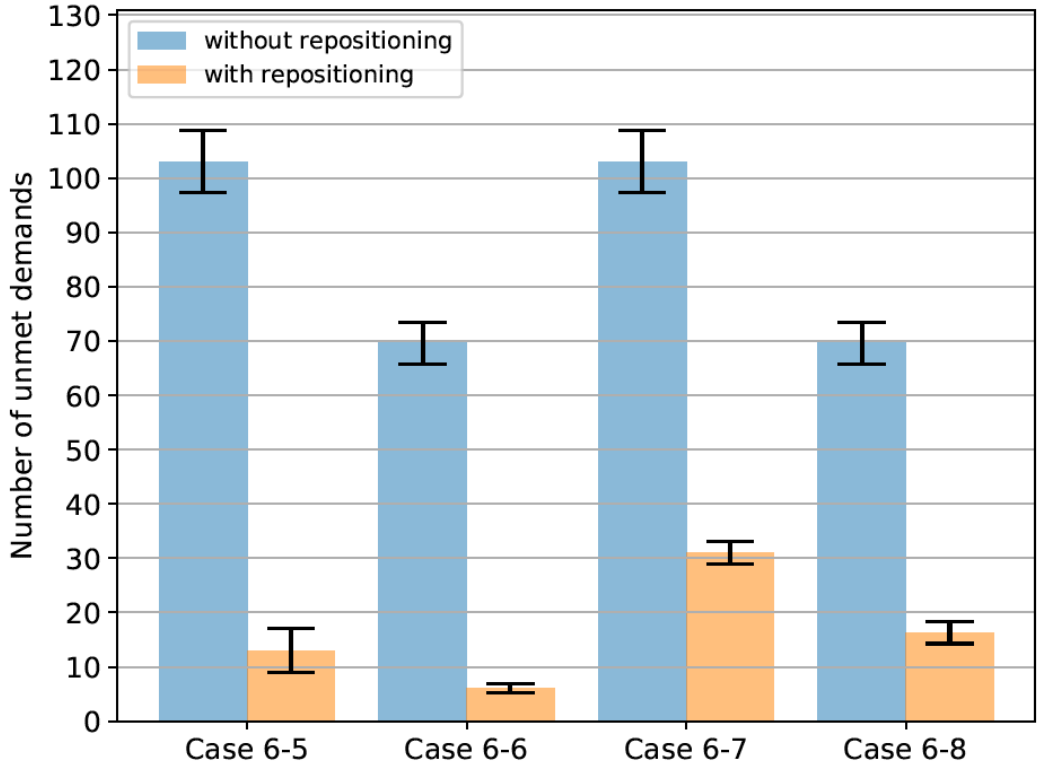


Figure 6-4 Weekday Afternoon Peak Numerical Experiments

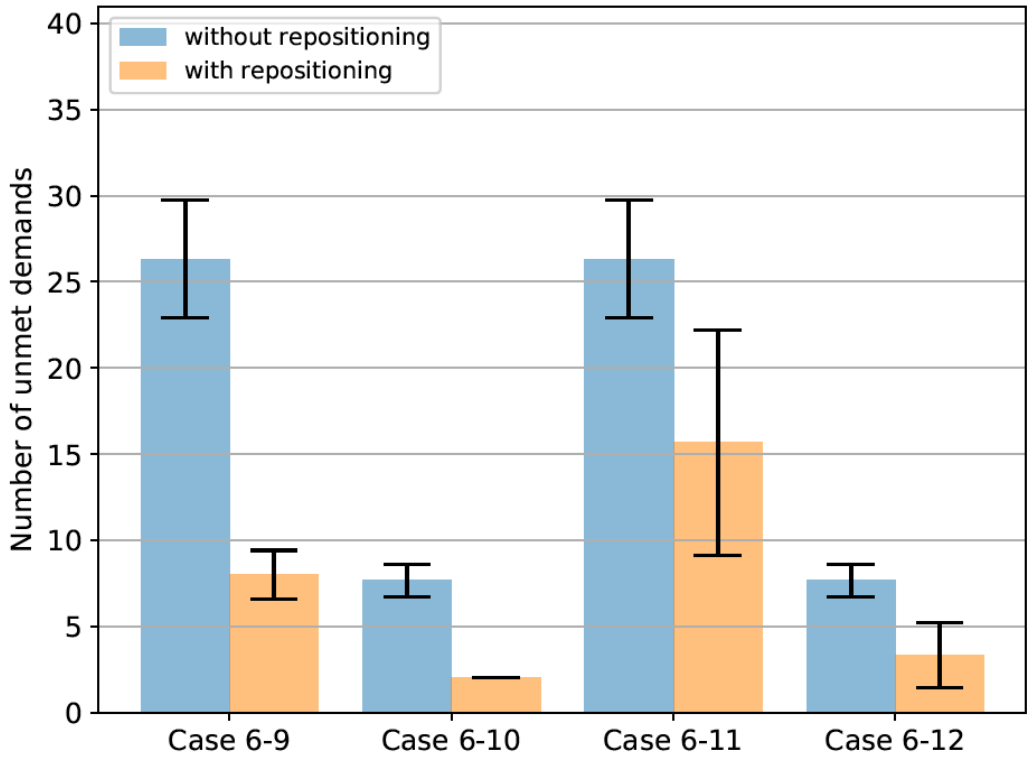


Figure 6-5 Weekend Mid-day Peak Numerical Experiments

Table 6-3 shows the percentage of reduction in unmet demand with repositioning. As expected, increasing the interval duration will result in a decline in the performance of the optimization module as the assumptions of the optimization module deviate from what actually holds in the real world. The average improvement compared to the case that there is no repositioning in the system is 84% and 69% for 30-minute and 60-minute intervals, respectively.

Table 6-3 Percentage Reduction in Unmet Demand

30 Minute		**60 Minute**	
Numerical experiment	**Mean percentage reduction in unmet demand**	**Numerical experiment**	**Mean percentage reduction in unmet demand**

Case 6-1	87%	**Case 6-3**	75%
Case 6-2	95%	**Case 6-4**	80%
Case 6-5	87%	**Case 6-7**	70%
Case 6-6	91%	**Case 6-8**	77%
Case 6-9	70%	**Case 6-11**	42%
Case 6-10	74%	**Case 6-12**	54%

Assumptions mentioned in section 3.1.1 can result in inconsistencies between the number of unmet demands expected by the repositioning plan and the result observed by running the simulation. Table 6-4 shows the comparison between the number of unmet demand expected by the repositioning plan and simulation in detail. For one of the instances of the first numerical experiment (case 6-1), the output of the optimization module expects a total of 2 unmet demands for the studied period, whereas the simulation module shows 6 unmet demands.

Table 6-4 Comparing Unmet Demand by Optimization Module and Simulation Module for One Instance of Case 6-1

Time	**Number of unmet pick-ups (optimization)**	**Number of unmet pick-ups (simulation)**	**Number of unmet drop-offs (optimization)**	**Number of unmet drop-offs (simulation)**
6:00-6:30	2	2	0	0
6:30-7:00	0	1	0	1
7:00-7:30	0	0	0	0

7:30-8:00	0	2	0	0
8:00-8:30	0	0	0	0
8:30-9:00	0	0	0	0
9:00-9:30	0	0	0	0
9:30-10:00	0	0	0	0

While the optimization module, by itself, produces optimistic solutions, and the solution quality often degrades when the decisions of the optimization module are simulated using the simulation module, compared to the case without repositioning, the model does indeed improve the simulated system considerably. In Table 6-5, the percentage differences in unmet demands between the optimization module and simulation module for the studied cases are summarized to showcase this. The quantity in Table 6-5 measures the following:

$$\text{Percentage difference in unmet demand} = \frac{z_{sim,opt} - z_{opt}}{z_{sim,None}} * 100 \qquad (6.1)$$

where z_{opt} is the unmet demand expected by the optimization module (i.e., the output of the optimization module), and $z_{sim,opt}$ is the unmet demand resulting from running the solution of the optimization module through the simulator. $z_{sim,None}$ is the unmet demand reported by the simulator assuming "no repositioning". Based on this metric, a smaller percentage difference in unmet demands is desirable. Note that percentage difference in unmet demand can be small (i.e., close to 0) either if the expected demand of the optimization module is very close to the result reported by the simulator, or if it

is non-zero but "no repositioning" would result in a large unmet demand (i.e., the denominator is large). From Table 6-5, it can be seen that relative to the unmet demand in the case without repositioning, the value of repositioning measured after the simulation is around 12% for the 30-minute cases and 14% for 60-minute cases.

Table 6-5 Percentage Difference in Unmet Demand between Optimization Module and Simulation Module

30 Minute		**60 Minute**	
Numerical experiment	**Mean percentage difference in unmet demand**	**Numerical experiment**	**Mean percentage difference in unmet demand**
Case 6-1	6%	**Case 6-3**	5%
Case 6-2	5%	**Case 6-4**	6%
Case 6-5	3%	**Case 6-7**	6%
Case 6-6	8%	**Case 6-8**	9%
Case 6-9	23%	**Case 6-11**	29%
Case 6-10	26%	**Case 6-12**	31%

6.4 *Effect of Adding Future Steps*

Cases 6-1, 6-2, 6-5, 6-6, 6-9, and 6-10 are resolved based on a 1-hour planning horizon (2 30-minute intervals) instead of 2 hours (4 30-minute intervals) to see the effect of multi-step planning in reducing the number of unmet demands and the routing cost. Figure 6-6 compares the number of unmet demands for these cases. On average, adding

1 hour to the planning horizon will result in around 97% reduction in the number of unmet demands.

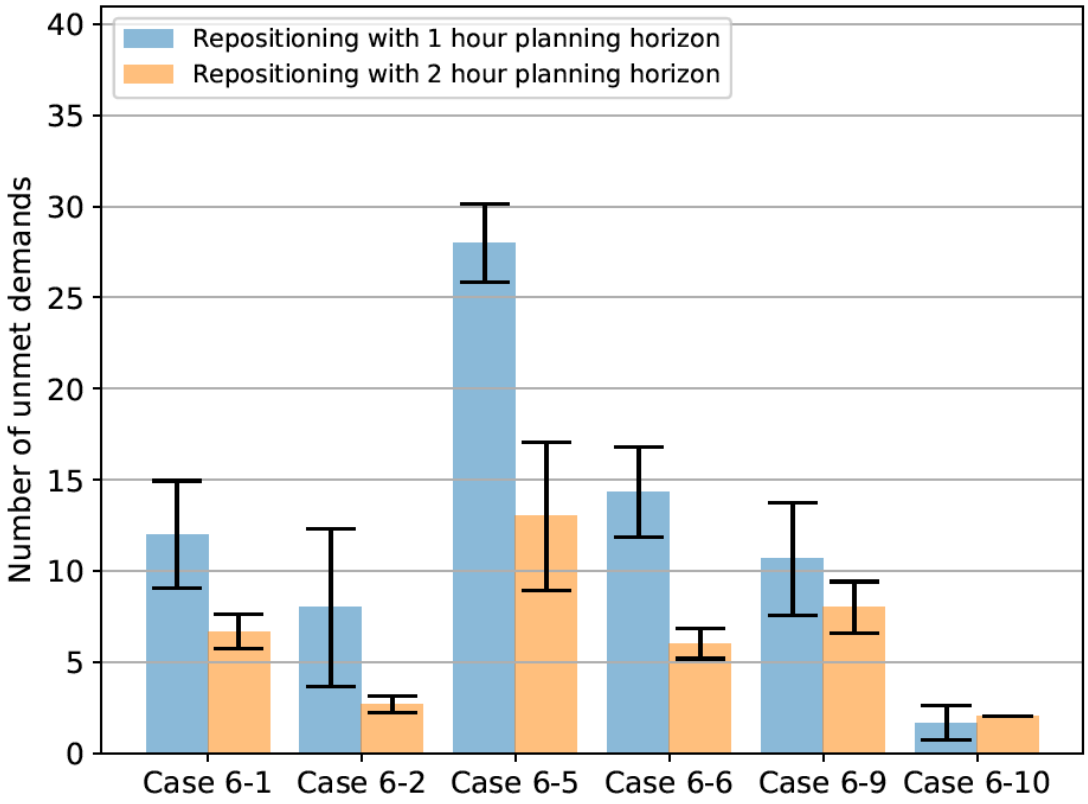


Figure 6-6 Number of Unmet Demands Versus Planning Horizon

Figure 6-7 compares the routing cost of cases when the planning horizon is fixed to 1 hour to when the planning horizon is fixed to 2 hours. On average, adding 1 hour to the planning horizon will result in around 20% reduction in the total time of routing. Overall, the results indicate accounting for demands in the future steps will result in significant reductions in the number of unmet demands as well as routing costs.

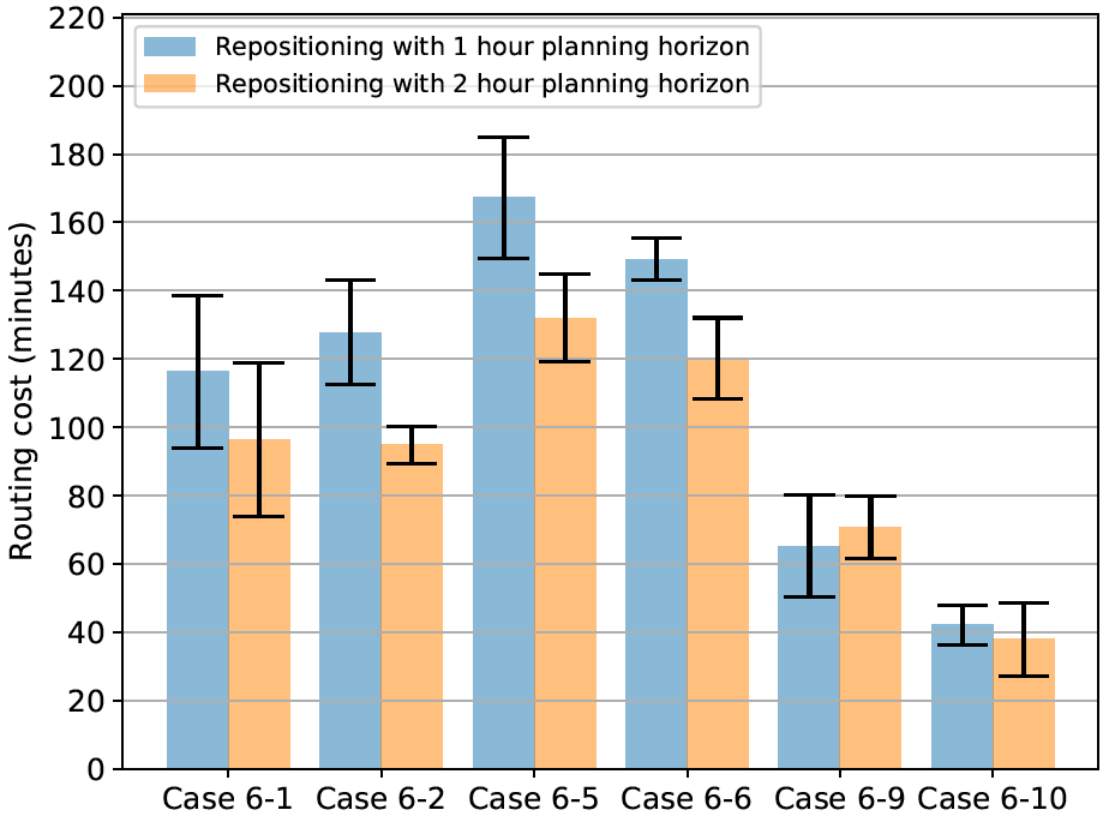


Figure 6-7 Routing Cost Versus Planning Horizon

6.5 *Running Time*

Although the results of section 6.3 show a significant reduction in the unmet demand of the bikeshare system by running the optimization module, the sensitivity analysis of running time versus the problem size on 16 cases (see Figure 6-8) shows that the proposed formulation can only solve problems with less than 20 stations in a reasonable time for a dynamic repositioning. A heuristic solution method is proposed to achieve a solution for this problem in a reasonable time. This method is described in the next chapter.

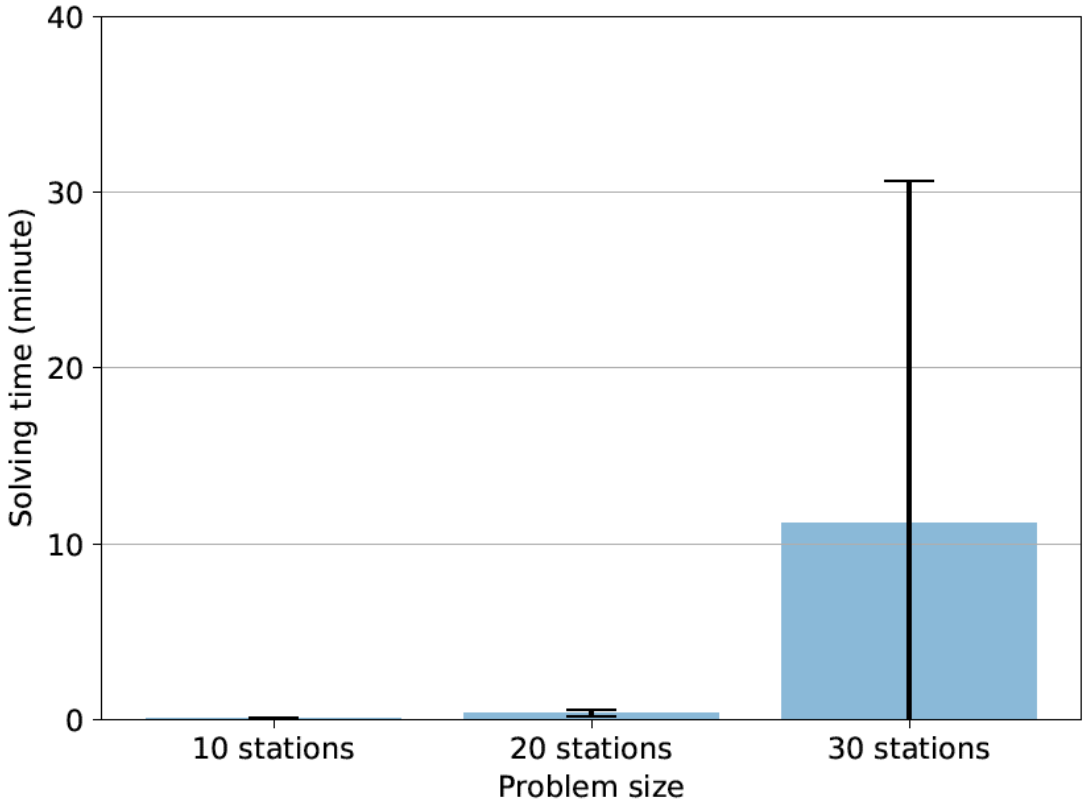


Figure 6-8 Solving Time Versus Problem Size

Chapter 7: Heuristic Method

7.1 *Overview*

Solving a repositioning problem even for one time interval is an NP-hard problem. The proposed heuristic, which aims to reduce the solution time of the optimization module, breaks the problem into three parts. The first part is generating source and demand pairs based on future pick-ups and drop-offs for the entire planning horizon. In the second and third parts, a cluster-first-route-second approach is utilized, where, in the second part, the next interval pairs are clustered and assigned to trucks. Finally, in the third part, a routing problem is solved for each cluster. In the following sections, each of these parts is explained in detail. Figure 7-1 shows the workflow of the heuristic method.

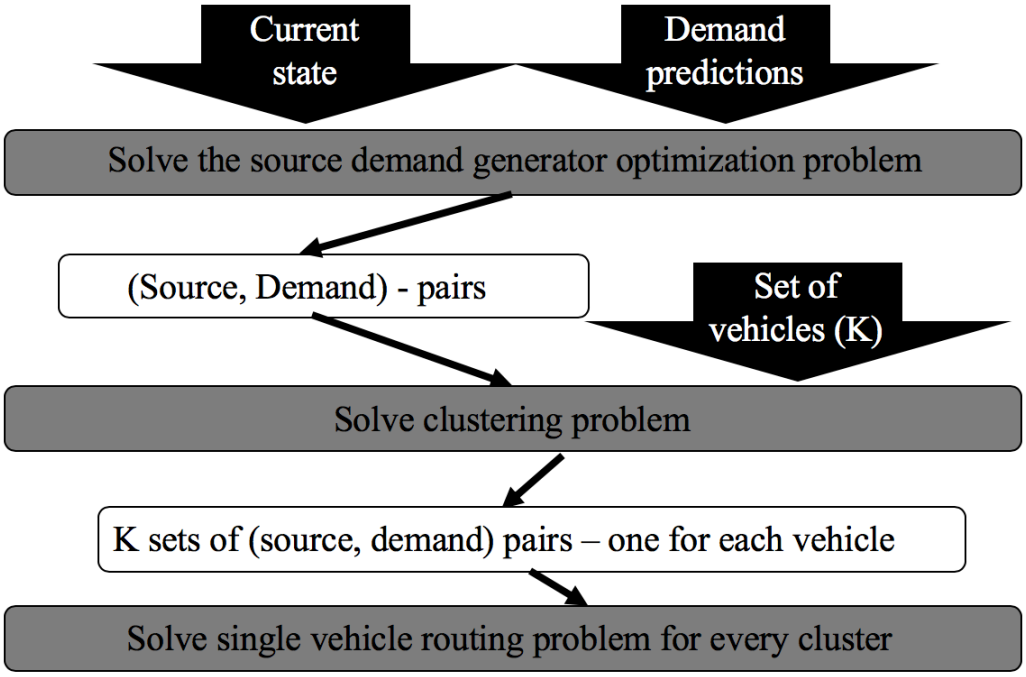


Figure 7-1 Heuristic Method Workflow

7.2 *Generating Source-demand Pairs*

The first step of the heuristic method is to generate the source and demand pairs for the movement of bikes for future planning time intervals. A MIP model is developed to find the optimal movements.

7.2.1 Notation

The following decision variables are used in addition to sets, parameters, and decision variables defined in section 3.2.1.

- Decision Variables

$b_{ss'}^{t}$ Binary indicator of traveling from the station (s) to station (s') at a time interval (t)

$l_{ss'}^{t}$ Number of bikes moved from the station (s) to station (s') at a time interval (t)

7.2.2 Objective Function

Equation (7.1) shows the objective function of the model. Similar to equation (3.1), the objective function of this model includes lost demand penalty (number of unmet demands), routing penalty, and initial trip penalty. Instead of having an exact routing penalty, a surrogate function is used for the routing penalty, which only accounts for the routing cost between the pick-up nodes (source nodes) and drop-off nodes (demand nodes).

$$Minimize\ Z = \sum_{t\in T}\sum_{s\in S}(pp_s^t + dp_s^t)\ + \alpha \cdot \sum_{t\in T}\sum_{s\in S}\sum_{s'\in S/\{s\}} b_{ss'}^t . TT_{ss'}^t + \beta \cdot \sum_{t\in T}\sum_{s\in S}\sum_{s'\in S/\{s\}} l_{ss'}^t \quad (7.1)$$

7.2.3 Constraints

In addition to constraints (3.28) to (3.38), the following constraints are included in the source-demand generator model.

Constraints (7.2) are very similar to constraints (3.21), and relate the available number of bikes at station s at time t+1 (a_s^{t+1}) to the available number of bikes at station s at time t (a_s^t) and the number of actual pick-ups (ap_s^t) and drop-offs (ad_s^t) at the station during the interval (t), and the number of bikes moved from other stations ($l_{s's}^t$) to this station and the number of bikes moved from this station to other stations ($l_{ss'}^t$).

$$a_s^{t+1} = a_s^t - ap_s^t + ad_s^t + \sum_{s'\in S/\{s\}} l_{s's}^t - \sum_{s'\in S/\{s\}} l_{ss'}^t \qquad \forall\, t \in T, s \in S \quad (7.2)$$

Constraints (7.3) and (7.4) limit the number of actual pick-ups and drop-offs.

$$ap_s^t \le ad_s^t + a_s^t - \sum_{s'\in S/\{s\}} l_{ss'}^t \qquad \forall\, t \in T, s \in S \quad (7.3)$$

$$ad_s^t \leq R_s - a_s^t + ap_s^t - \sum_{s' \in S/\{s\}} l_{s's}^t \qquad \forall t \in T, s \in S \qquad (7.4)$$

Constraints (7.5) enforce the load moved from s to s' ($l_{ss'}^t$) to be zero if the binary indicator of traveling from s to s' ($b_{ss'}^t$) is 0. In other words, we can move bikes only if the truck is traveling the path.

$$l_{ss'}^t \leq C * b_{ss'}^t \qquad \forall t \in T, s\&s' \in S \qquad (7.5)$$

Constraints (7.6) to (3.13) are similar to constraints (3.11) to (3.18).

$$\sum_{s' \in S/\{s\}} l_{ss'}^t \leq aux1_s^t \qquad \forall t \in T, s \in S \qquad (7.6)$$

$$a_s^t - ap_s^t + M * R_s * baux1_s^t \geq 0 \qquad \forall t \in T, s \in S \qquad (7.7)$$

$$aux1_s^t \leq M * R_s * (1 - baux1_s^t) \qquad \forall t \in T, s \in S \qquad (7.8)$$

$$aux1_s^t \leq a_s^t - ap_s^t + M * R_s * baux1_s^t \qquad \forall t \in T, s \in S \qquad (7.9)$$

$$\sum_{s' \in S/\{s\}} l_{ss'}^t \leq aux2_s^t \qquad \forall t \in T, s \in S \qquad (7.10)$$

$$R_s - a_s^t - ad_s^t + M * R_s * baux2_s^t \geq 0 \qquad \forall t \in T, s \in S \qquad (7.11)$$

$$aux2_s^t \leq M * R_s * (1 - baux2_s^t) \qquad \forall\, t \in T, s \in S \qquad (7.12)$$

$$aux2_s^t \leq R_s - a_s^t - \ ad_s^t + M * R_s * baux2_s^t \qquad \forall\, t \in T, s \in S \qquad (7.13)$$

Constraints (7.14) and (7.15) indicate that we cannot move any bikes or add bikes to a station if that station does not have enough bikes for pick-ups or enough racks for drop-offs.

$$\sum_{s' \in S/\{s\}} l_{ss'}^t \leq (1 - \ bpp_s^t) * R_s \qquad \forall\, t \ \in T, s \in S \qquad (7.14)$$

$$\sum_{s' \in S/\{s\}} l_{s's}^t \leq (1 - \ bdp_s^t) * R_s \qquad \forall\, t \ \in T, s \in S \qquad (7.15)$$

7.3 *Clustering*

The proposed clustering algorithm is similar to bottom-up hierarchical clustering. The inputs of the algorithm are the number of clusters/trucks (K), location of trucks (L_k), and source-demand pairs (source and demand stations' location) generated (SD) using the source-demand generator model that was explained in the previous section. The output of the clustering model is a set of pairs that need to be covered by each truck (CR).

The clustering algorithm is summarized in Figure 7-2.

Figure 7-2 Proposed Clustering Algorithm

Input: K, L_k, SD

Output: CR

1. $CR \leftarrow SD$

2. $N \leftarrow |CR|$

3. **while** |K| $< N$: [Reduce the size of the output set to $|K|$] **do**

    3.1. Calculate/update the pair-wise distance between all (source, demand) pairs in CR

    3.2. $(s_l, d_l), (s_n, d_n) \leftarrow$ Find the two (source, demand) pairs which have the minimum distance

    3.3. $(s_m, d_m) \leftarrow$ Merge the two (source, demand) pair into a super-node

    3.4. $CR \leftarrow CR/\{(s_l, d_l), (s_n, d_n)\}$[Remove the two pairs from the output set]

    3.5. $CR \leftarrow CR \cup \{(s_m, d_m)\}$[Add the two merged pair to the output set]

    3.6. $N \leftarrow N - 1$ [Decrement the size of the output set by 1]

4. Solve an assignment problem to assign every element in $|K|$ to every element in $|CR|$ based on distance between location of trucks (L_k) and super-nodes.

7.4 *Routing Model*

7.4.1 Overview

The output of the source-demand generator model and clustering model is the subset of stations that need to be covered by each truck and their corresponding number of

bikes that need to be picked up or dropped off. To find the order of visiting the nodes, we need to develop a routing model for the generated pick-up and drop-off pairs. The routing model only develops routing plans for the next interval and not all the intervals of the rolling horizon. This allows for taking advantage of the new information and updating the routing model with the new predictions for future intervals. A modified version of the dial and ride problem proposed by Cordeau (2006) is used for the routing model

Similar to pick up and delivery problems, the routing model is defined on a directed graph, where, N_k is the vertex set partitioned into $\{P, D, \{0\}\}$, $P = \{1, \ldots, n\}$ is the set of pick-up stations (i.e., stations for which a truck needs to pick up bikes from), $D = \{n+1, \ldots, 2n\}$ is the set of corresponding drop-off stations, and 0 is an auxiliary station where the truck is located at the beginning/end of the interval and has a distance of zero to all other stations. Number of pick-ups and drop-offs (Q_s) are also the output of source-demand generator model and $Q_s = -Q_{s-n}$.

7.4.2 Notation

The following parameters and decision variables are used in addition to sets, parameters, and decision variables defined in section 3.2.1.

- Sets

S_{sub}	The subset of stations that need to be covered by the truck
N_k	Set of stations + auxiliary origin/destination node (artificial)
	$N_k = S_{sub} \cup \{0\}$

- Parameters

Q_s Number of bikes loaded/unloaded at the station (s)

ω Weight of routing penalty

SS Start station

- Decision Variables

$qtot_s$ Number of bikes in the truck after visiting station (s)

$x_{ss\prime}$ Binary indicator for the truck traveling from the station (s) to the station (s')

b_s Binary indicator for visiting station (s)

y_s Subtour elimination variables

7.4.3 Objective Function

Unlike most routing models that try to minimize the routing cost, this model has the objective of maximizing the number of served demands. Equation (7.16) shows the objective function of the model, which has two parts. The first part maximizes the served demands, and the second part minimizes the routing cost. Routing duration/cost is a secondary objective. The routing cost is included with a small coefficient (ω) to avoid picking schedules with high routing costs among the ones that have the same number of served demands.

$$Maximize\ Z = \sum_{s \in S_{sub}} b_s \cdot Q_s - \omega \cdot \sum_{s \in S_{sub}} \sum_{s\prime \in S_{sub}/\{s\}} x_{ss\prime} \cdot TT_{ss'} \quad (7.16)$$

7.4.4 Constraints

Every truck, as mentioned, needs to start from the auxiliary origin, visit the start node (SS), and end at auxiliary destination for each interval. Constraints (7.17) and (7.18) make sure these constraints are met.

$$x_{0SS} = 1 \tag{7.17}$$

$$\sum_{s \in S_{sub}} x_{s0} = 1 \tag{7.18}$$

Constraints (7.19) ensure that stations are only served if the truck departs the auxiliary station.

$$\sum_{s\prime \in S_{sub}/\{s\}} x_{ss\prime} \geq b_s \qquad \forall s \in S_{sub} \tag{7.19}$$

Constraints (7.20) are the conservation of flow constraints

$$\sum_{s\prime \in S_{sub}/\{s\}} x_{ss\prime} = \sum_{s\prime \in S_{sub}/\{s\}} x_{s\prime s} \qquad \forall s \in S_{sub} \tag{7.20}$$

Constraints (7.21) are constraints on the load after visiting a given station s.

$$\max(0, Q_s) \leq qtot_s \leq \min(C, \mathrm{C} + qtot_s) \qquad \forall s \in S_{sub} \tag{7.21}$$

Constraints (7.22) are for conservation of load, which are linearized in constraints (7.23).

$$qtot_s \geq (qtot_{s\prime} + Q_s) * x_{s'_s} \qquad \forall s \in S_{sub} \qquad (7.22)$$

$$qtot_s \geq qtot_{s'} + Q_s - \mathrm{C} * (1 - x_{s'_s}) \qquad \forall s \in S_{sub} \qquad (7.23)$$

As noted in 7.4.1, stations are named so that $s \in \{1, \ldots, n\}$ are the set of pick-up stations and $s \in \{n+1, \ldots, 2n\}$ are the set of corresponding drop-off stations. Constraints (7.24) make sure that both source and demand are served for each pair of source and demand generated by the source-demand generator model.

$$b_s = b_{s+n} \qquad \forall s \in S_{sub} \qquad (7.24)$$

The routing model also includes subtour elimination constraints, which look similar to those included in the complete optimization model (equation (3.39) presented in section 3.2.1.

$$y_s - y_{s\prime} + (|\mathrm{N}| - 1) * x_{s'_s} \leq (|\mathrm{N}| - 2) \qquad \forall s \in S_{sub}, \forall s' \in S_{sub}/\{s\} \qquad (7.25)$$

7.5 *Beam Search*

Due to the overall nature of the source-demand generator model, solving the source-demand generator model will likely result in a multi-optimal solution. For example, in many cases, there is a possibility to move the bikes from a particular station to another station either at this period or in a future period. Given that the main problem is a multi-

period problem, different solution among the multiple optimal solutions of the first subproblem (i.e., source-demand generator), can result in different solutions with varying qualities for the subsequent parts/problems. A beam search algorithm is developed to choose the most promising solution at each interval by inspecting a predetermined number of optimal solutions generated from the source-demand generator model. The advantage of this approach is that once the source-demand generator model is solved to optimality, finding few additional alternate optimal solutions incurs little to no additional cost. Also, the rest of the steps (including clustering and routing) can be solved in parallel for each of the solutions as they are independent. After solving all of the branches, a criterion is used for choosing the best solution. Here, $\sum_{s\in S_{sub}} b_s \cdot Q_s / \sum_{s\in S_{sub}} \sum_{s\prime\in S_{sub}/\{s\}} x_{ss\prime} \cdot TT_{ss\prime}$ which is the number of moved bikes divided by the routing duration is chosen to evaluate different solutions. The solution with the highest ratio is picked as the most promising solution for each interval and fed to the simulation module for evaluation.

7.6 *Numerical Results*

In this section, several case studies are developed for comparing the results of the complete model and the heuristic model. Figure 7-3 to Figure 7-5 show the location of the stations of the numerical experiments, and

Table 7-1 summarizes the characteristics of the numerical experiments.

All the numerical experiments are solved with three different initial values for the availability of bikes for each station. The number of initial bikes in the stations are

randomly chosen, such that the ratio of the total number of bikes over the capacity is between 45% to 55%. Repositioning is done using trucks with a capacity of 20 bikes, and the starting locations of the trucks are randomly chosen for each case study.

For all the cases, α and β are set to 0.01, and for the heuristic method ω is set to 0.01. The interval duration is set to 30 minutes, and the planning horizon is set to 2 hours. The effect of interval duration is discussed in section 7.6.2.

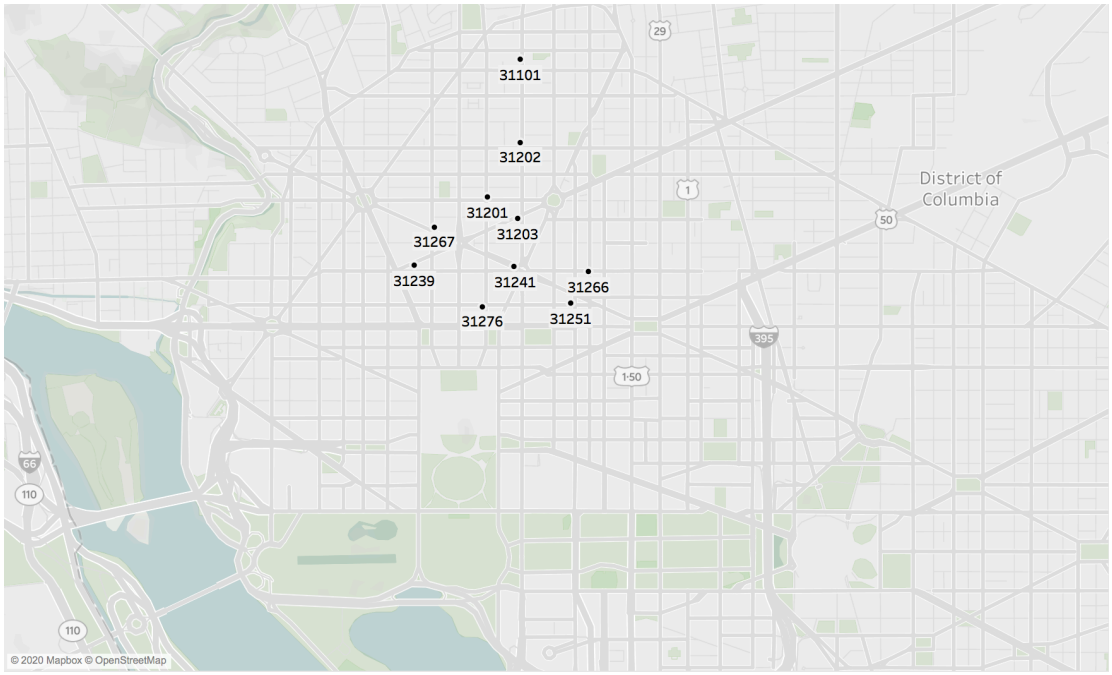


Figure 7-3 Location of Stations for Case 7-1 (10 Stations)

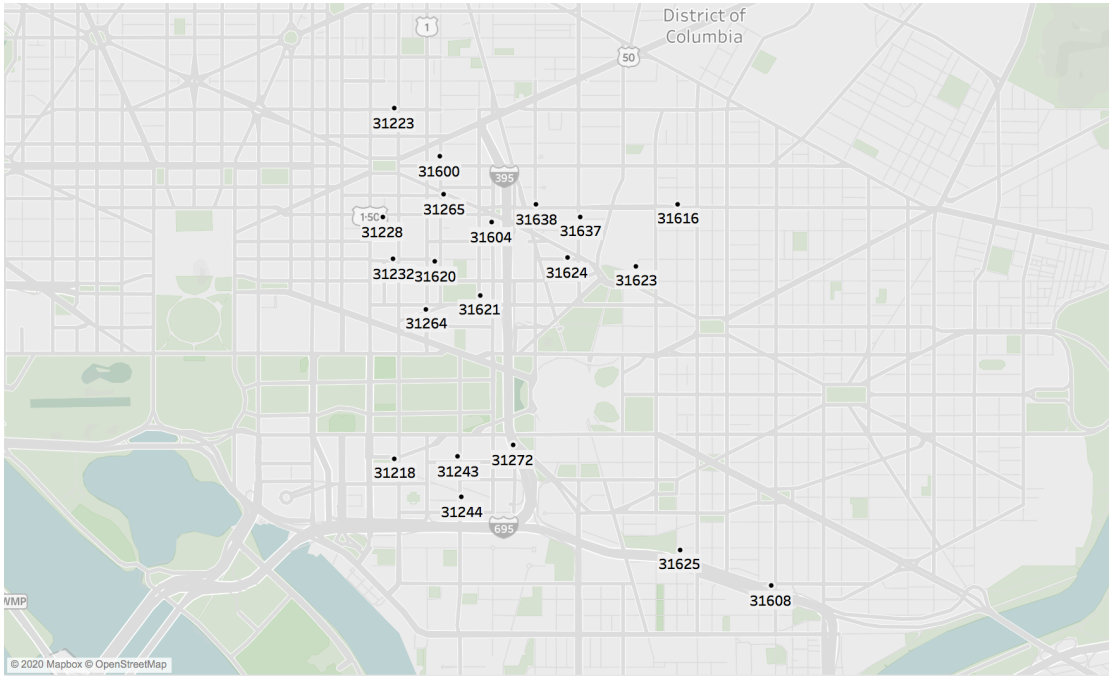


Figure 7-4 Location of Stations for Case 7-2 (20 Stations)

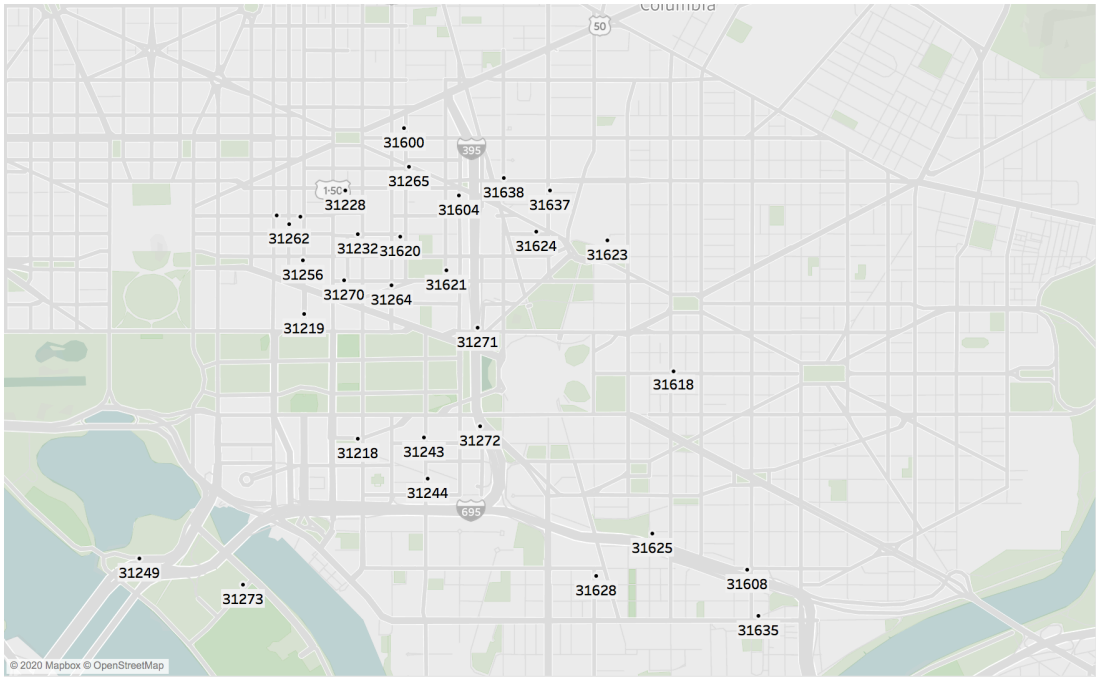


Figure 7-5 Location of Stations for Case 7-3 (30 Stations)

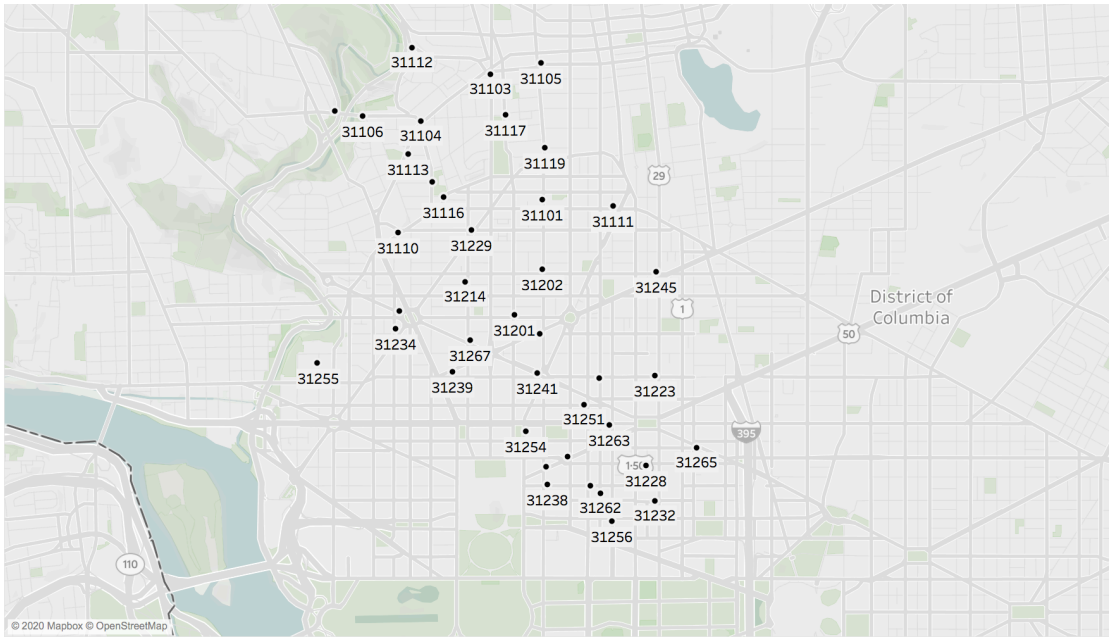


Figure 7-6 Location of Stations for Case 7-4 (40 Stations)

Table 7-1 Characteristics of the Numerical Experiments

Numerical experiment	**Time of day**	**Number of stations**	**Interval duration**	**Number of trucks**	**Total demand**
Case 7-1	weekday afternoon peak	10	30 minute	1	588
Case 7-2	weekday afternoon peak	20	30 minute	1	1248
Case 7-3	weekday afternoon peak	30	30 minute	1	1779
Case 7-4	weekday morning peak	40	30 minute	2	1136

7.6.1 Number of Unmet Demands and Routing Cost Results

For each numerical experiment, the five following models are compared in terms of number of unmet demands and routing cost (time):

a) Model 0: Running the simulation without any repositioning to see the number of unmet demands without repositioning.
b) Model 1: Running the complete optimization model discussed in chapter 5 with the assumption that we know the number of pick-ups and drop-offs for each station during each time interval (P_s^t and D_s^t).
c) Model 2: Running the heuristic model discussed in this chapter with the assumption that we know the number of pick-ups and drop-offs for each station during each time interval.
d) Model 3: Running the heuristic model discussed in this chapter with the predictions from the prediction model proposed in Chapter 5.
e) Model 4: Running the heuristic model discussed in this chapter with the predictions from a naïve prediction method. The naïve prediction method uses the number of pick-ups and drop-offs of the station during the same weekday and time of day of the previous week.

Model 0 gives us a baseline for assessing the value of repositioning. The repositioning can be done based on model 1 or model 2. By comparing model 2 to model 3, we can evaluate the value of having perfect information. By comparing model 3 and model 4, we can appraise the proposed prediction model by comparing it with a naïve but strong prediction baseline.

Figure 7-7 and Figure 7-8 show the number of unmet demands and routing times of the described models (model 0 to model 4) for the case 7-1, respectively.

Both the complete model (model 1) and the heuristic method (model 2) have average unmet demand of around 2 (a 96% reduction of unmet demand compared to model 0) which illustrates that the heuristic is performing on par with the complete model when evaluated using the number of unmet demands. However, when comparing model 1 and model 2 in terms of routing cost, the complete model (model 1) has a shorter routing time. The average routing time of heuristic is around 50 minutes, whereas the routing time of the complete model is around 35 minutes. Model 3 reduces the unmet demand around 80%, whereas model 4 only reduces the unmet demand around 51% compared to the model without any repositioning (model 0), which shows the value of the prediction model.

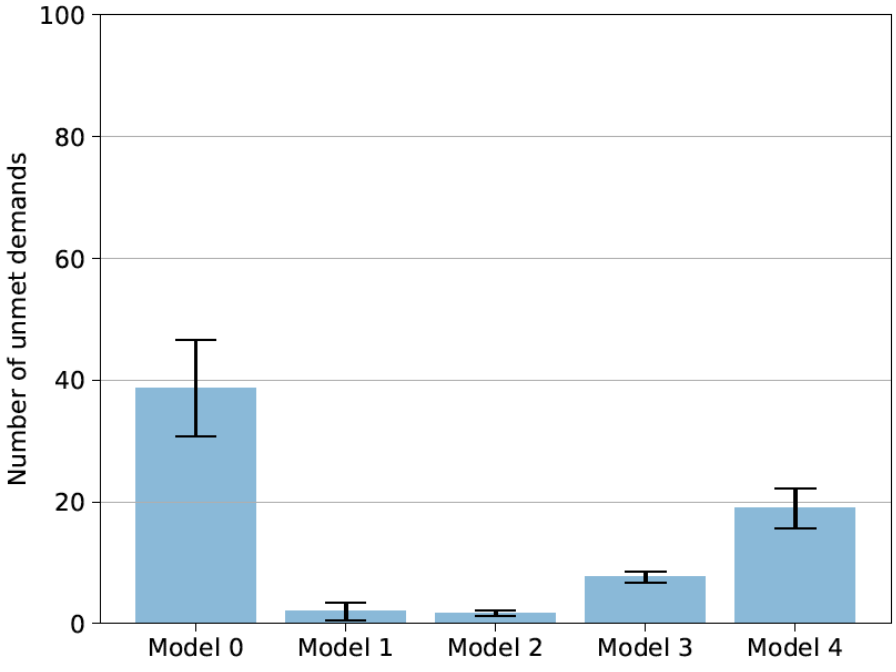


Figure 7-7 Comparison of Different Models' Unmet Demand for Case 7-1

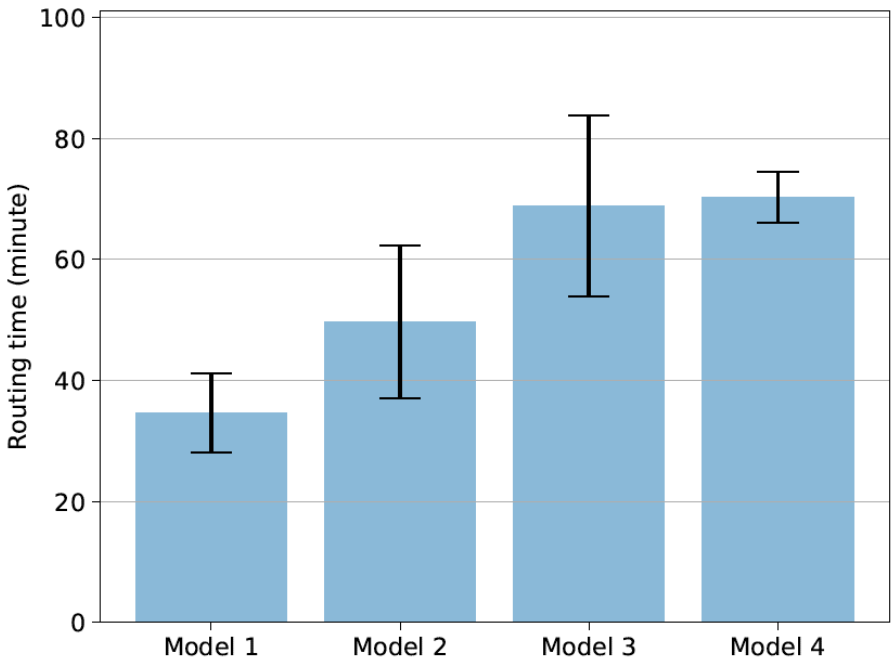


Figure 7-8 Comparison of Different Models' Routing Time for Case 7-1

The number of unmet demands and routing times for the different models evaluated using case 7-2 are shown in Figure 7-9 and Figure 7-10, respectively. Similar to case 7-1, on average, model 1 and model 2 have similar performance in terms of unmet demand. However, the routing time of model 2 is around 12% higher than model 1. Also, for this case, model 3 has a slightly lower number of unmet demands compared to model 4; however, the routing time of model 4 is 32% higher than model 3.

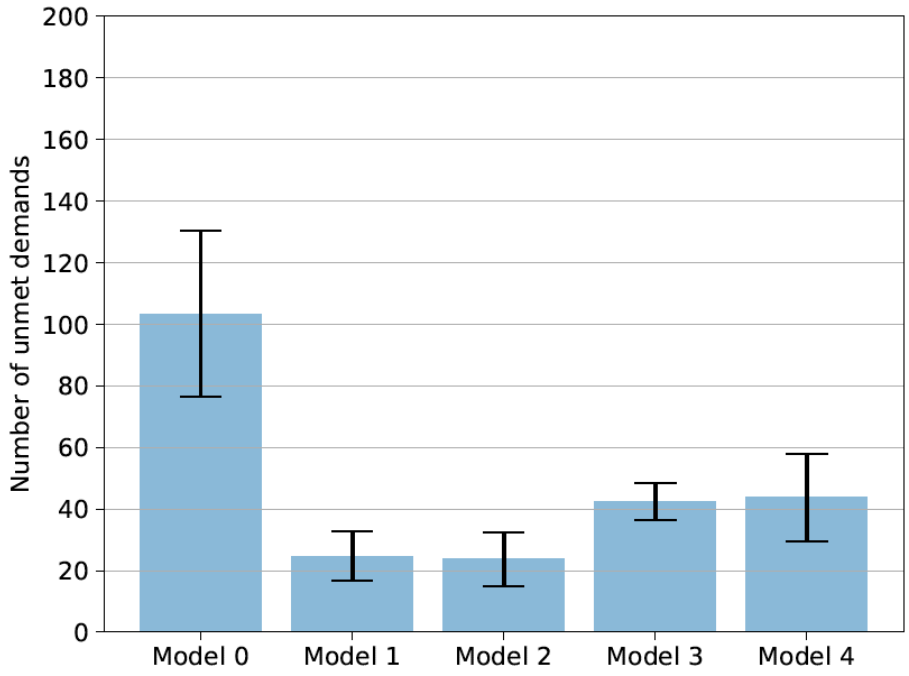


Figure 7-9 Comparison of Different Models' Unmet Demand for Case 7-2

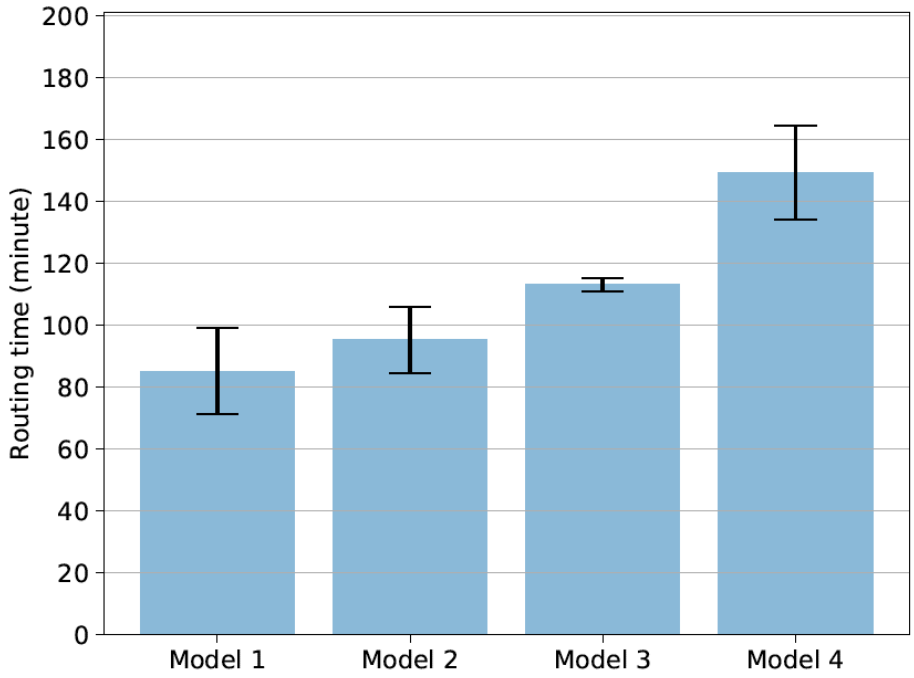


Figure 7-10 Comparison of Different Models' Routing Time for Case 7-2

Figure 7-11 and Figure 7-12 show the number of unmet demands and routing times for case 7-3. Results indicate by using the proposed prediction model and heuristic method

(model 3), we can reduce the unmet demand by around 62% compared to the baseline model 0. If we had a prediction model with the capability of predicting the exact number of pick-ups and the number of drop-offs, we could reduce the unmet demand to around 83% (using model 2). Comparing the routing time of model 2 and model 3 in Figure 7-12, we can conclude that there is not going to be a significant reduction in the routing time by having a prediction model with high accuracy.

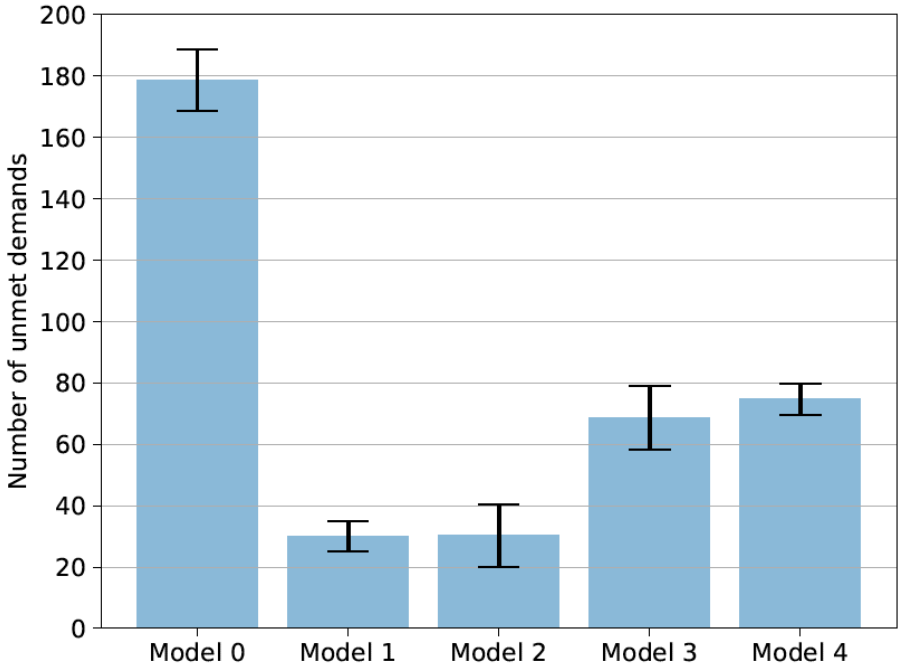


Figure 7-11 Comparison of Different Models' Unmet Demand for Case 7-3

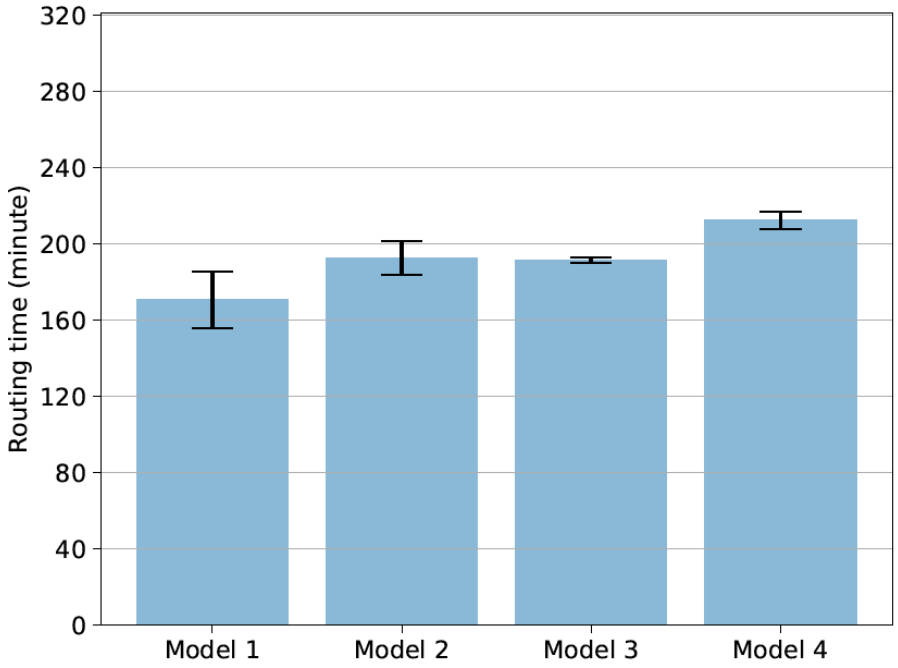


Figure 7-12 Comparison of Different Models' Routing Time for Case 7-3

Figure 7-13 shows the unmet demand for case 7-4. For this case, due to the problem size, the complete model did not find a solution even after running the model for more than 12 hours, so we could not present the results of this model (model 1).

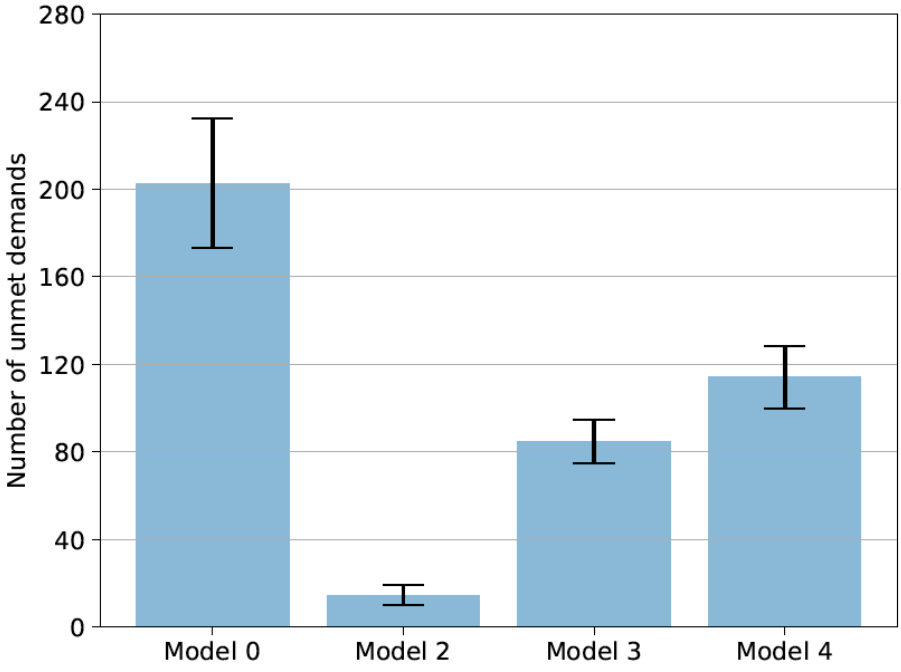


Figure 7-13 Comparison of Different Models' Unmet Demand for Case 7-4

Figure 7-14 shows the routing time of model 2, 3, and 4. The routing time of models 2 and 3 are similar to each other and slightly higher than that of model 4.

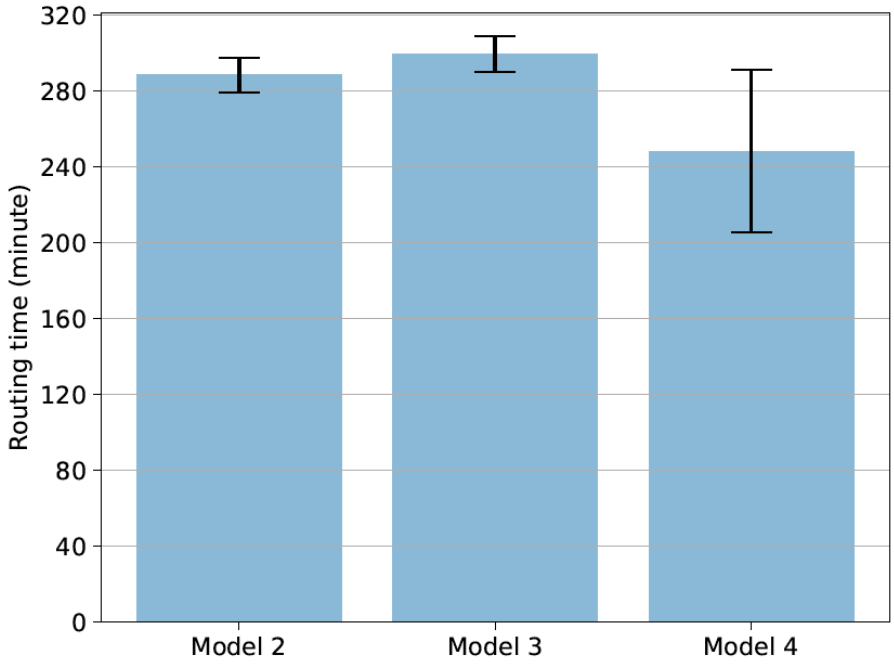


Figure 7-14 Comparison of Different Models' Routing Time for Case 7-4

Overall, the results of this section show that the quality of solutions of the heuristic method (model 2) is comparable to that of the complete model (model 1). Although the routing time of the heuristic method is slightly higher than the complete model, In all of the cases, the two methods have similar unmet demands.

The model with the trained prediction model (model 3) has lower unmet demand in all of the cases tested compared to model 4, which uses the naïve method for predicting the number of pick-ups and drop-offs. Model 3 can reduce the unmet demand by up to 30% compared to model 4. The average reduction is around 14%. Also, in all of the cases except 7-4, model 4's routing time is higher than that of model 3.

Results also indicate using the proposed heuristic model in conjunction with the trained prediction model (model 3) can reduce the unmet demand up to 85% compared to the baseline model without repositioning (model 0). Using model 3, the average reduction in unmet demand is 64%. For model 2, the average reduction is around 87%. This suggests improving the prediction model can potentially reduce the unmet demand by 23% for the cases tested.

7.6.2 Effect of Interval Duration

As discussed in section 3.1.1, and further illustrated using the numerical results in section 6.3, the assumptions of the optimization module hold better for shorter time intervals. However, the prediction power of the model has the opposite relation with respect to shorter time intervals (see section 5.4.2). The case studies in

Table 7-1 are resolved using 60-minute interval durations instead of the 30-minute interval duration to see the overall performance of the model with respect to the interval durations. We use model 3 (heuristic method with predicted demands) for comparisons.

Figure 7-15 and Figure 7-16 compare the routing cost, and unmet demand of the model with 30 minutes interval duration, and the model with 60 minutes interval duration, respectively.

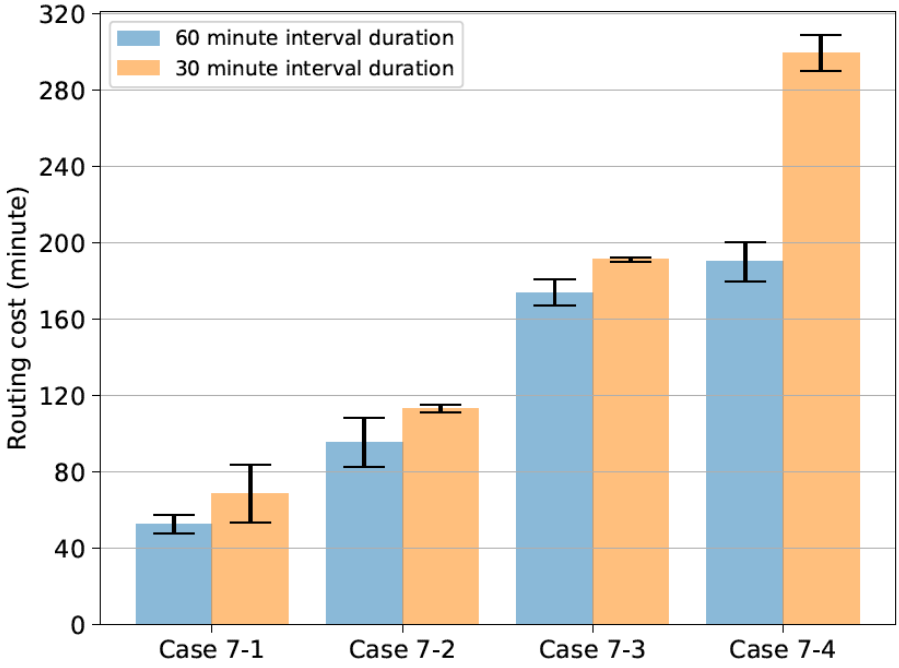


Figure 7-15 Routing Costs of 30 and 60 Minutes Interval Durations

Although in all the cases the routing time of the model with the 60-minute interval duration is lower, the unmet demand of the model with 30 minutes is significantly lower (see Figure 7-16). The average reduction in the unmet demand is around 48% for the model with 60 minutes interval, whereas this reduction is around 64% for the model with 30 minutes interval.

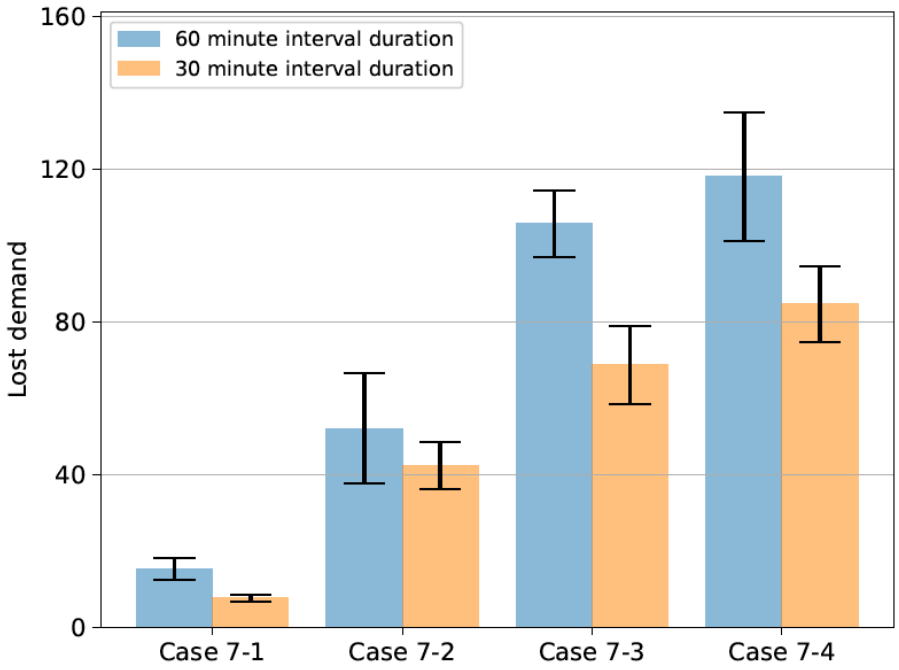


Figure 7-16 Number of Unmet Demands of 30 and 60 Minutes Interval Durations

7.6.3 Running Time

Table 7-2 to Table 7-5 summarize the summary statistics of the running time of the optimization module for each of the numerical experiments. Each of the numerical experiments in

Table 7-1 is run for a specific time period (for the definition of time periods, see chapter 6). The min, max, median, and mean running time of the interval durations (30 min) within each time period are reported in Table 7-2 to Table 7-5. Similar to previous experiments, results are averaged across all the different initializations for the availability of bikes. Comparing model 2 with all the instances solved by the heuristic

method (except case 7-4[4]) suggests that model 2 can find a good solution faster (up to 44 times faster) than model 1.

It is worth noting that the heuristic method (model 2) can be even further accelerated by reducing the problem size using data preprocessing. One possible method for this preprocessing is explained in Chapter 8.

Table 7-2 Comparison of Different Models Running Time for Case Study 7-1

Model	**Max (Sec)**	**Min (Sec)**	**Median (Sec)**	**Mean (Sec)**	**Total time (Sec)**
Model 1	1.9	0.1	0.4	0.6	4.9
Model 2	0.3	0.2	0.1	0.2	1.4
Model 3	0.4	0.1	0.2	0.2	1.5
Model 4	0.5	0.1	0.1	0.2	1.5

Table 7-3 Comparison of Different Models Running Time for Case Study 7-2

Model	**Max (Sec)**	**Min (Sec)**	**Median (Sec)**	**Mean (Sec)**	**Total time (Sec)**
Model 1	41.1	2.3	8.2	12.7	102
Model 2	0.7	0.1	0.3	0.3	2.6
Model 3	0.6	0.1	0.3	0.3	2.4
Model 4	4.1	0.2	0.4	1.0	7.6

[4] For case 7-4, model 1 (the complete model) was not able to find a solution even when it was executed for 12 hours.

Table 7-4 Comparison of Different Models Running Time for Case Study 7-3

Model	**Max (Sec)**	**Min (Sec)**	**Median (Sec)**	**Mean (Sec)**	**Total time (Sec)**
Model 1	4794.3	62.3	235.8	890.5	7123.6
Model 2	50.1	1.1	9.4	16.3	159.3
Model 3	3.1	0.7	1.3	1.5	12.0
Model 4	11.4	0.6	3.2	4.0	31.8

Table 7-5 Comparison of Different Models Running Time for Case Study 7-4

Model	**Max (Sec)**	**Min (Sec)**	**Median (Sec)**	**Mean (Sec)**	**Total time (Sec)**
Model 1	-	-	-	-	-
Model 2	99.1	2.5	14.9	24.5	196.3
Model 3	43.7	1.9	9.4	13.7	109.5
Model 4	25.9	1.5	7.9	9.6	76.7

Chapter 8: Case Study

The heuristic method, combined with the prediction model, is tested on one week of Capital Bikeshare data during different time periods. Similar to previous numerical experiments, time periods considered are weekday morning peak (6 AM – 10 AM) and weekday afternoon peak (3 PM – 7 PM).

The predictions are for four 30-minute future intervals, and the planning horizon of the optimization module is four (2 hours). The repositioning is done using trucks with capacity 20. The availability of the bikes for each station is initialized by random so that the total number of bikes in the system is within 45% – 55% of the total number of docks.

The number of stations considered in the case study is 208. There are some stations that are somewhat balanced. This observation can be used to prune the problem size. For example, the stations for which the prediction model predicts they have the same number of pick-ups and drop-offs may require fewer visits by trucks. While these stations may not have any shortage of demands among themselves, visiting them can still be helpful for limiting the shortage of demands at other stations (i.e., moving bikes from one of these stations to a station which has a shortage of bikes.) The inclusion of these stations is always good as long as it does not hurt running time. Due to this trade-off, one can have a fixed number of stations to be used in the heuristic in mind. The stations which are included in the subset of stations used in the heuristic can be evaluated and sorted based on their importance. One possible metric for importance is as follows which computes the "imbalanceness" of a station s:

$$I_s = \sum_{t\in\{1,2,3,4\}} \max\{\,0,\quad -\left(D_{s,t} - P_{s,t} + A_{s,t}\right),\ \ D_{s,t} - P_{s,t} + A_{s,t} - C_s\}$$

where $D_{s,t}$, and $P_{s,t}$ are the drop-off and pick-up predicted by the prediction model for station s at time t, $A_{s,t}$ is the availability of bikes of station s at time t, and C_s is the capacity of station s (i.e., the number of docks of the station.) The above metric is non-negative and will be equal to zero for the stations which might not incur a demand penalty based on the prediction model. If the fixed number of stations is more than the number of stations with $I_s > 0$, then some of the stations with $I_s = 0$ could also be included at random.

8.1 *System Evaluation without Rebalancing*

As a base-line, the system is simulated without rebalancing. For the considered morning peaks (i.e., different weekdays), no rebalancing results in 679 to 1058 unmet demands. And for the considered afternoon peaks, the unmet demands will be 567 to 805. To put things in perspective, the morning peaks' total number of pick-ups and drop-offs are between 4,107 and 5,461, and the afternoon peaks' total is between 6,565 and 7,574.

8.2 *System Evaluation with Heuristic and Effect of the Number of Vehicles*

As stated before, the repositioning of problems with more than 20 stations could only be done using the heuristic method. Also, given that the problem size of the final case study is large compared to the illustrative numerical studies of the previous chapters, this case study can be used for evaluating the value of having additional trucks in the fleet. As can be seen in Figure 8-1, for the morning peaks, the heuristic is able to find

solutions that result in 55% fewer unmet demands. Adding more trucks can result in solutions with fewer unmet demands. However, going beyond more than 5 trucks for this case study does not considerably decrease the number of unmet demand but considerably increases the routing time of the trucks, as seen in Figure 8-2, which results in a drop in efficiency of trucks. A comprehensive sensitivity analysis can be used by operators of bikesharing systems to make decisions regarding the truck fleet size.

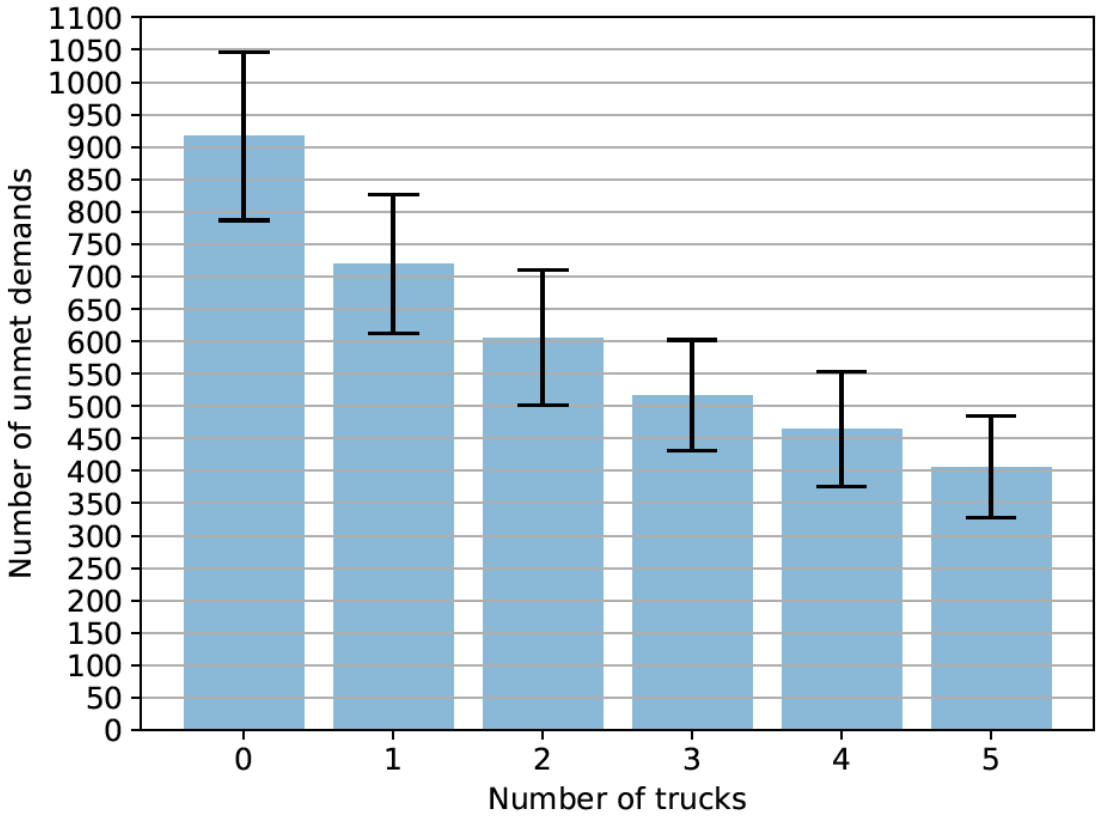


Figure 8-1 Effect of Number of Trucks on Unmet Demand for A.M. Peak

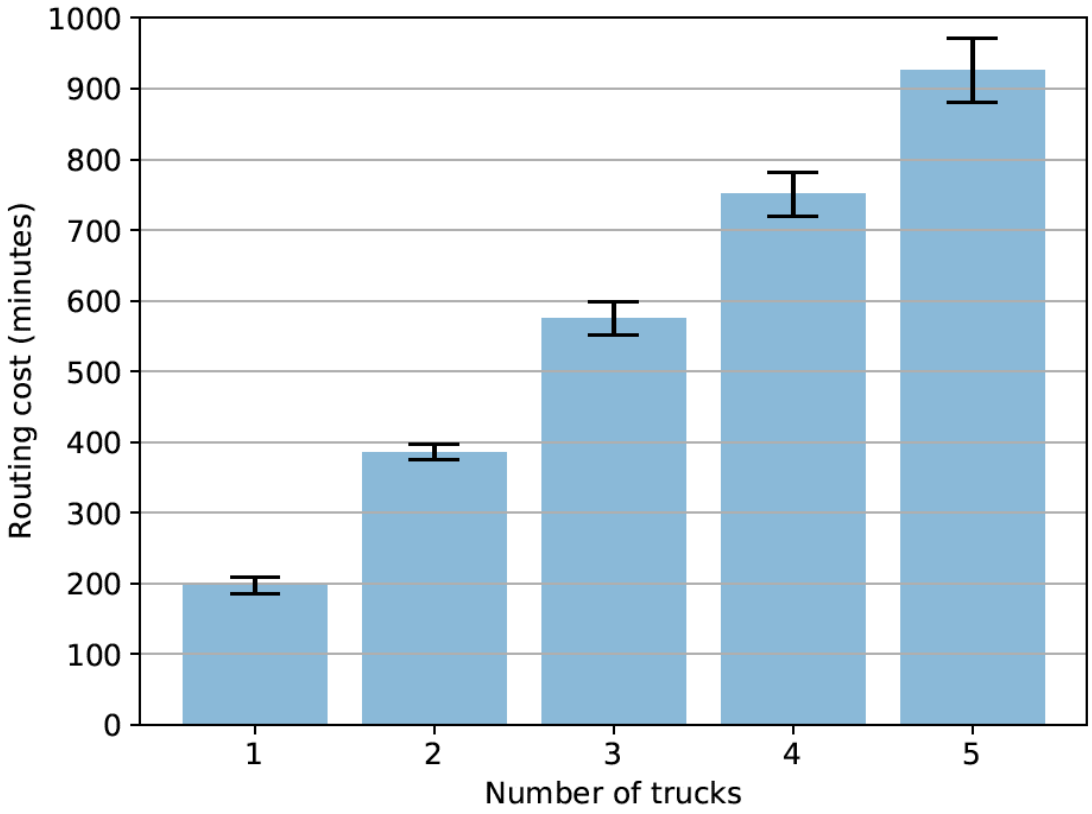


Figure 8-2 Effect of Number of Trucks on Total Routing Time for A.M. Peak

Similar results for the afternoon peak can be observed in Figure 8-3 and Figure 8-4. The number of unmet demands can approximately be cut in half using the combined heuristic and prediction model.

Overall, results indicate that for this particular case study, the bikeshare system is more balanced during P.M. peaks compared to morning peaks as the average unmet demands without repositioning is around 900 for A.M. peaks, whereas for P.M. peaks this number is around 700.

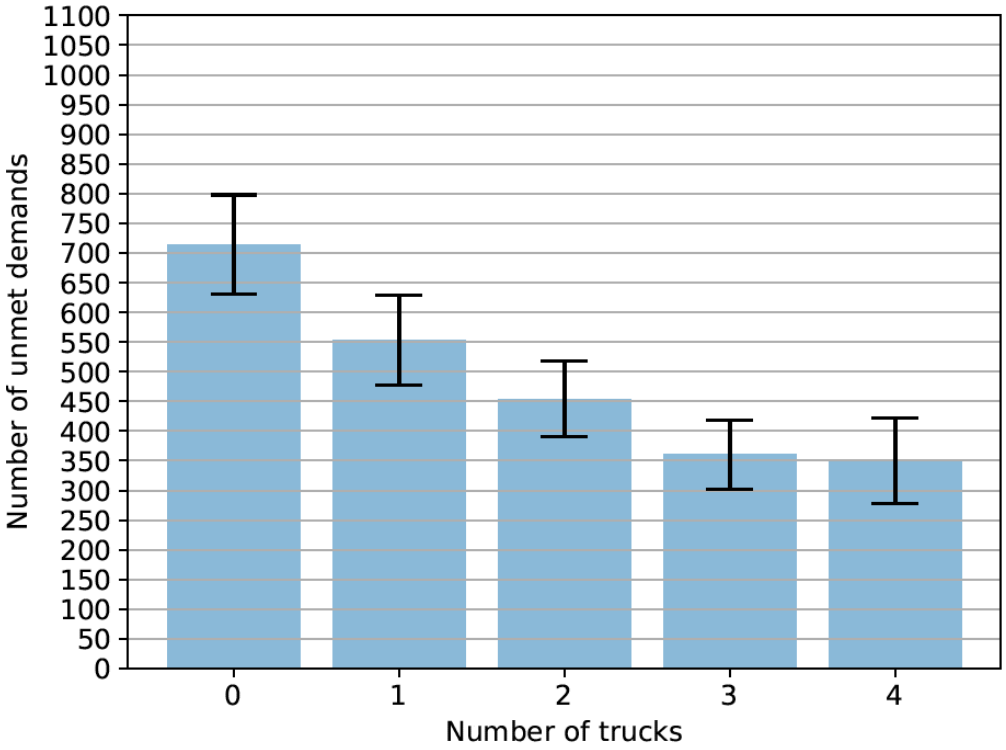


Figure 8-3 Effect of Number of Trucks on Unmet Demand for P.M. Peak

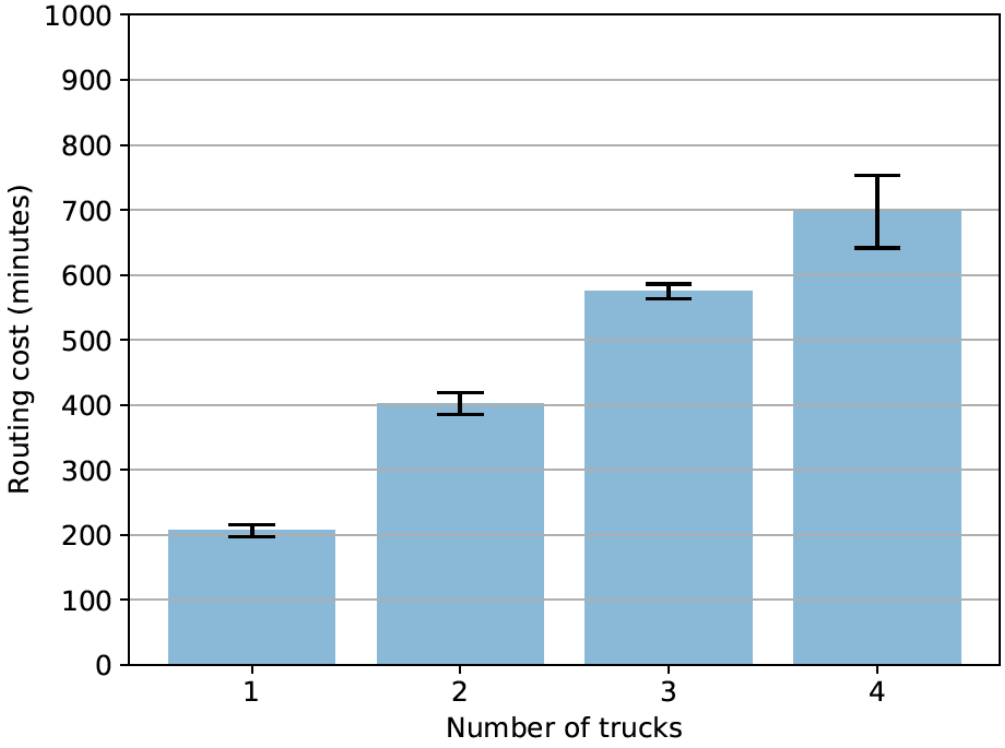


Figure 8-4 Effect of Number of Trucks on Total Routing Time for P.M. Peak

Chapter 9: Summary, Conclusions, and Future Research

9.1 *Summary*

In this study, an integrated prediction and multi-period optimization model was proposed for solving the repositioning problem for bikesharing systems during peak periods.

In the context of prediction models, several prediction models from the literature, namely, dynamic regression, random forests, and feedforward neural networks, were compared. For the purpose of predicting station-level pick-ups and drop-offs of the Capital Bikeshare data, the feedforward neural network resulted in the best performance. Consequently, feedforward based regression models were utilized as base models of the prediction model.

Moreover, the inclusion of membership information and station status information were proposed as possibly useful sources of information which can potentially improve prediction models' performance. Membership information is included in the meta-data of each trip, and status information is available per station. The status of stations includes information on the duration of the outage (full or empty). Through visualizing the user behavior of members and non-members (casual users), it can be observed that these categories of users make use of the bikeshare systems in different ways. This observation is consistent with previous works.

In the context of the optimization model, a MIP model was proposed. The optimization model is a multi-period model with a rolling horizon. Several assumptions were made

to model the system. These assumptions were evaluated using a discrete-event simulation module. The complete MIP model results in good solutions but does not scale well to larger problems with more stations. A three-step heuristic model was proposed, which allows the operators to solve the rebalancing problem for real-world size instances. The solutions found using the heuristic method decrease the unmet demand by a large factor. By performing an analysis of the real-world problem, the system operators can assess the value of having additional trucks.

9.2 *Conclusions*

The results of the prediction model confirmed that adding the status of stations as an input to the prediction model as well as having separate prediction models for the number of pick-ups (number of drop-offs) of casual users and member will improve the quality of predictions in terms of root mean squared error. Results show that adding this information can improve the root mean squared error of the prediction model by up to 30%. The median improvement is around 19%.

A comparison of the multi-step model to the model without any consideration for future demands shows that accounting for future demands can significantly improve the efficiency of the repositioning plan. Results of studied cases indicate adding 1 hour to the planning horizon will result in around 97% reduction in the number of unmet demands and a 20% decrease in the routing time duration.

Limited sensitivity analysis on the interval duration suggests the quality of the prediction model improves by increasing the duration of the time intervals (due to an increase in the signal/noise ratio); however, the assumptions of the optimization

module do not hold very well when the interval duration increases. Results of studied cases on two different interval durations show that the overall performance of the integrated model is better when the interval duration is shorter.

The prediction module, when combined with the optimization module, resulted in system improvements compared to the case where the prediction module is replaced with reasonable temporal-based baselines such as the naïve temporal method, which predicts that the demand is similar to the same time and day of the previously observed week. On average, the combined model reduces the unmet demand by around 14% compared to the model that uses the naïve temporal method. However, comparing the combined model with the model that uses the actual number of pick-ups and drop-offs suggests that the repositioning model can still benefit from improvements to the prediction model.

9.3 *Future Research*

The proposed models and methods in this dissertation have some assumptions and limitations which can be relaxed for future studies. Some possible directions for future research are:

- In this dissertation, the value of information on the user's membership and the status of stations is evaluated on the forecasting power of the prediction model. Although these findings are not model specific, it is of interest to compare and evaluate this forecasting power using other existing models in the literature. As the numerical results suggest, the efficiency of the proposed model highly depends on the quality of predictions. Among other models in the literature and not tested

in this study, Graph Convolutional Networks and especially Spatio-Temporal Graph Convolutional Networks (STGCN) has shown promising results in tackling time series prediction on graph-structured data (Zhang et al. 2019, Yu et al. 2017) and can be possibly tested to improve the predictions. Moreover, this study did not account for the effect of special events on the dynamics of the time series.

- Solutions from the proposed heuristic model could only be compared to the solutions of the complete model in cases with a limited number of stations. For larger cases, the complete model cannot find a feasible solution within a reasonable time limit. It is desirable to verify the quality of the solutions found by the heuristic with other new heuristic methods. Moreover, to further reduce the solution time of the complete model and parts of the heuristic that are solved by commercial solvers (such as the routing model), a branch and cut algorithm can be incorporated into the model.
- As the focus of the current study is on developing an operational model for repositioning bikes in a system during peak hours, there are limited numerical experiments on strategic level decisions related to the repositioning problem (e.g., number of trucks and their capacity). The presented model in this dissertation can be utilized for recommending strategic actions/decisions by performing extensive case-study experiments. Although we have tested the effect of using different initialization methods in the workload of repositioning during peak hours and the performance of the repositioning model, there is a need to further evaluate the option of rebalancing during non-peak hours with the objective of reducing the workload of repositioning during peak hours as well as evaluating the effect of

having different ratios for the number of bikes and capacity at each station. Moreover, the coefficients of lost demand penalty (quantity of unmet demands) and routing penalty can be adjusted in the objective function using data from the intended bikeshare system to maximize the profit. One way to this could be to set the coefficients of lost demand so that stations that tend to have a higher profit per pick-up or drop-off have larger penalties for their unmet demand as well.

- Starting from 2019, some of the bikeshare systems located in North America, including Capital bikeshare, have initiated pilot programs for introducing electric bikes into their station-based bikesharing systems. It is of interest to study how the rebalancing operations need to be possibly modified. For example, one might be interested in mostly positioning these bikes in stations along uphill routes where users need to bike uphill, and one might need to account for recharging and battery swapping of these bikes.

Appendix

Appendix A Model Comparison

A.1 Overview

Some results for prediction module based on dynamic regression models (DRM), random forest, and neural network models are presented here. To evaluate these results, they are compared with base models such as naïve and seasonal naïve.

Results for all the models are based on one-step forecasts without re-estimation, which means that the model is estimated based on a single set of training data, and the estimated model is used to forecast the number of pick-ups and drop-offs for the next 30-minute time interval for the test data.

The data is divided into two sets of training and testing data. The first training dataset is from January 1st to June 27th, 2017, and the testing dataset for this data is from June 28th to June 30th, 2017. The second training dataset is from January 1st to December 22nd, and the testing dataset is from December 23rd to December 25th, 2017. This allows us to have a better comparison by accounting for different time of year and consequently different levels of demands.

Four stations are selected from the 208 sampled stations. Their locations are shown in Figure A-1. To see the performance of the model on the stations with different usage (high, low, and medium demand), these four stations are chosen which have different levels of demand: Jefferson Dr. & 14th St SW as a high demand station, Eastern Market / 7th & North Carolina Ave SE and 4th & East Capitol St NE as two medium demand stations, and 19th & East Capitol St SE as a low demand station.

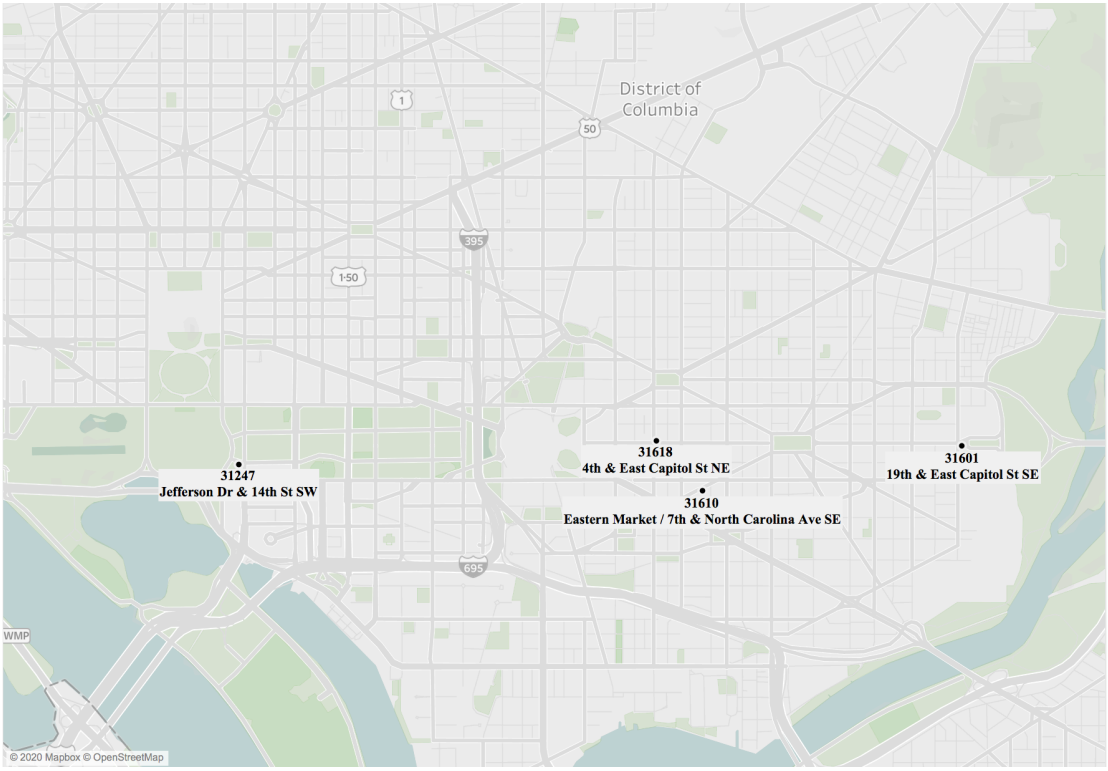


Figure A-1 Four Sampled Stations' Locations and Their IDs

The effects of the following variables are included in the models:

- Weather data, including precipitation data, and temperature data that is retrieved from NOAA (national oceanic and atmospheric administration)'s national centers for environmental information (NCEI). Two cases are considered for incorporating these features: (a) continuous features - these features are represented in the models as they are, and (b) categorical features - they are represented in a discrete and categorical form. Two different conversions to categorical variables are used. In the first one, numbers are converted based on equal size bins (7°F) for temperature and (0.05 inch) for precipitation. This results in 12 categories for temperature and nine categories for precipitation. In the second one, these categories are merged based on their median (categories with similar medians are merged into one category) to

get fewer categories and have more observations in each category. The second case results in nine categories for temperature and six categories for precipitation.

- Hour of the day, the day of the week, and monthly indicators are also added to the models. One-hot-encoding is used to include these categorical variables in the model. For the neural network and random forest, the previous observations are also added (These are automatically included for the DRM, so we do not need to add them as regressors.)

A.2 Naïve, Seasonal Naïve, and DRM Method Results

In the naïve method, all the forecasts are set to be equal to the last observed value. In the seasonal naïve method, the forecasts are set to be the last value observed from the same season of the year (here, the observation from the same day and interval of the last week is used).

To fit the DRM model to the data, the series are required to be stationary (stabilizing mean and variance), and then the DRM model will be used to fit the transformed stationary data. Box-cox transformation and differencing are two methods for stabilizing variance and mean. After this step, different models were developed by changing the order of the autoregressive part, the moving average part, and by including different regressors. The final models for each station are chosen based on the Akaike Information Criterion, and then the model is tested on the test data.

Based on the best results from the DRM, the demand is correlated with most of the considered/selected features, especially the day of the week, temperature, and time of day. Precipitation is not included in our final models. The impact of each of these terms is different for different stations and for pick-up/drop-off models. For example the

fitted model for "Jefferson Dr. & 14th St SW pick-ups" has a relatively large coefficient for time intervals that are within 12:30 P.M. to 5:30 P.M. Whereas for Eastern Market / 7th & North Carolina Ave SE station's pick-ups, these intervals are between 7:30 to 9:00 A.M. In general, the effects of days of the week and times of the day are different for different stations.

A.3 Random Forest Method Results

As it was discussed in the literature review (section 2.2.4), random forest fits a number of classifying decision trees by choosing random samples from the training data and randomly selecting a subset of features (regressors). Here, the minimum squared error is used to measure the quality of splits at the nodes of the tree. There are a few hyperparameters that require tuning for the construction of the trees. One is the minimum number of samples required to be at a leaf node. Four different numbers are tested for this parameter: 1, 2, 3, and 4. For the number of trees in the forest, five numbers are tested: 50, 100, 250, 500, and 1000. For the maximum number of features that can be used for constructing trees, two different numbers are tested: number of features itself and square root of the number of features. Temperature and precipitation are tested both in continuous and categorical formats, but for the final cross-validation, only categorical formats of temperature are used since this format results in higher accuracies. Similar to DRM, precipitation is removed from the candidate/initial features since it doesn't increase the accuracy of the model. As a result, for each station, and each training dataset, 80 different combinations of hyperparameters are tested using k-fold cross-validation. Here k is set to three. The final model for each station is

chosen based on the average performance on the folds, and then the best model is trained on the full training set and evaluated on the test set.

Feature importance is calculated using error reduction, and the number of samples at each internal node that splits on that feature. Overall, for all four stations, a general trend can be observed that previous observations, along with the temperature and day of the week, have higher importance compared to the other features. Similar to DRM, different stations have different hours of the day, which have the highest importance.

A.4 Multilayer Feedforward Neural Network (DNN) Results

Similar to the other approaches, DNNs have hyperparameters that require tuning. Unlike the previous methods though, DNNs have many hyperparameters. To restrict the search space, the architecture search space is limited to having at most two hidden layers. All architectures have one neuron as the output. Once a prediction is made, the continuous prediction is rounded to the closest integer value; then, this integral value is used to calculate the desired metric. Root mean squared error between the prediction and the true value is used as the loss function for training the DNN. To accelerate learning and as a regularization, batch normalization is included before the activations of every hidden layer. Other regularizers, including weight decay with a coefficient of 1e-4, are also included to prevent overfitting on the training data.

While the general architecture is similar to Figure A-2, the number of neurons for the first (n_{h_1}) and second hidden layer (n_{h_2}) are varied. In our experiments, $n_{h_1} \in \{30, 35\}$ and $n_{h_2} \in \{5,10,15,20\}$. An adaptive learning rate that decays if the loss of the model increases is used, but the initial learning rate is set to one of {0.01, 0.05, 0.1, 0.15}. Another hyperparameter that is considered is related to the stopping condition. The

number of epochs is set to be 500, 1000, or 2000. In the end, for each station and pick-up/drop-off, the best model based on the validation set is selected and is used for predicting the pick-up or drop-off of the test set.

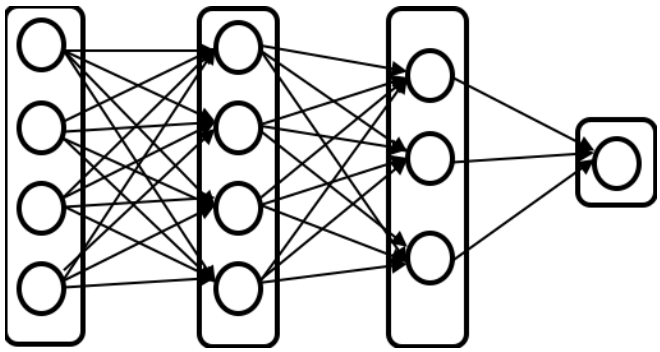


Figure A-2 General Architecture of DNNs with Two Hidden Layers.

Figure A-3 to Figure A-10 illustrate the predicted number of pick-ups and drop-offs by using the selected models of the DRM, random forest, and DNN versus actual pick-ups and drop-offs for the four stations under consideration, and for the first test dataset.

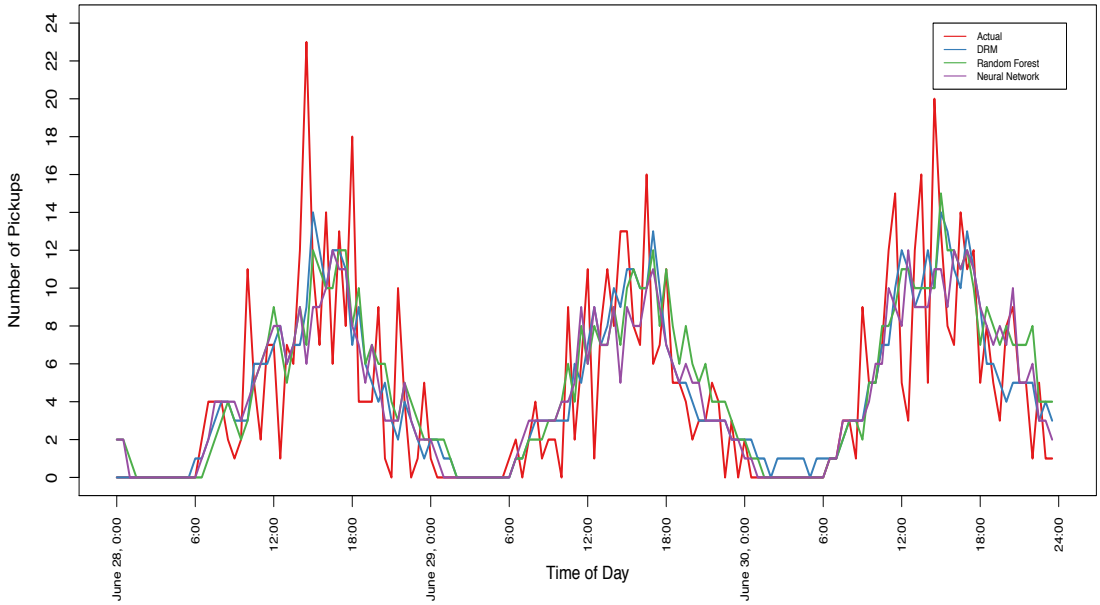


Figure A-3 Jefferson Dr. & 14th St SW the Predicted Number of Pick-ups Versus Actual Numbers

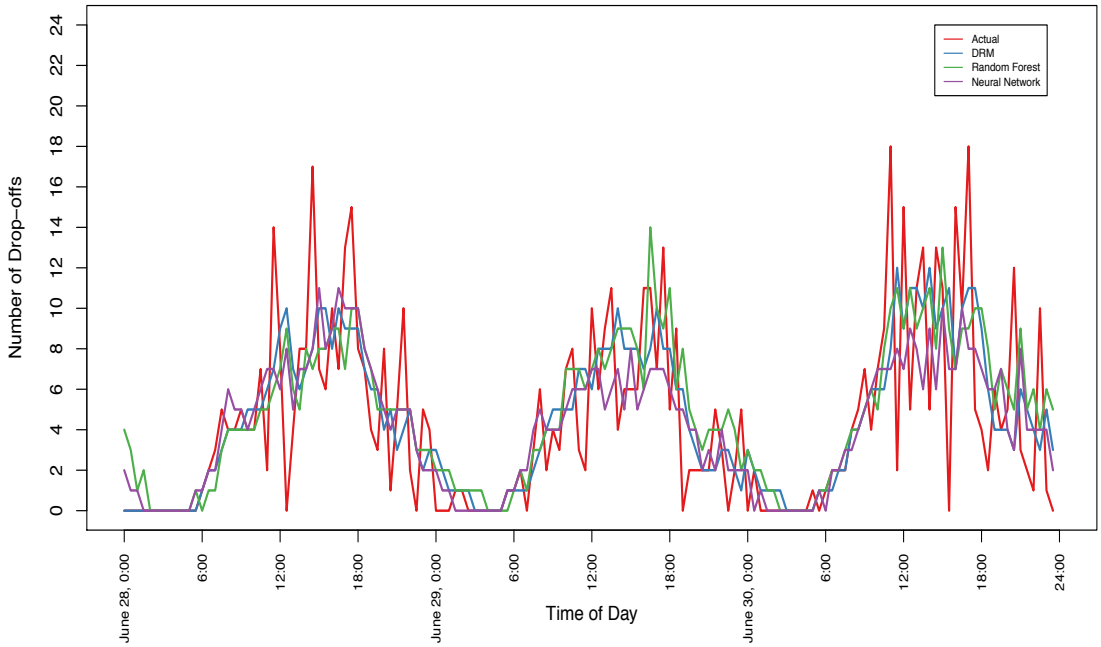


Figure A-4 Jefferson Dr. & 14th St SW the Predicted Number of Drop-offs Versus Actual Numbers

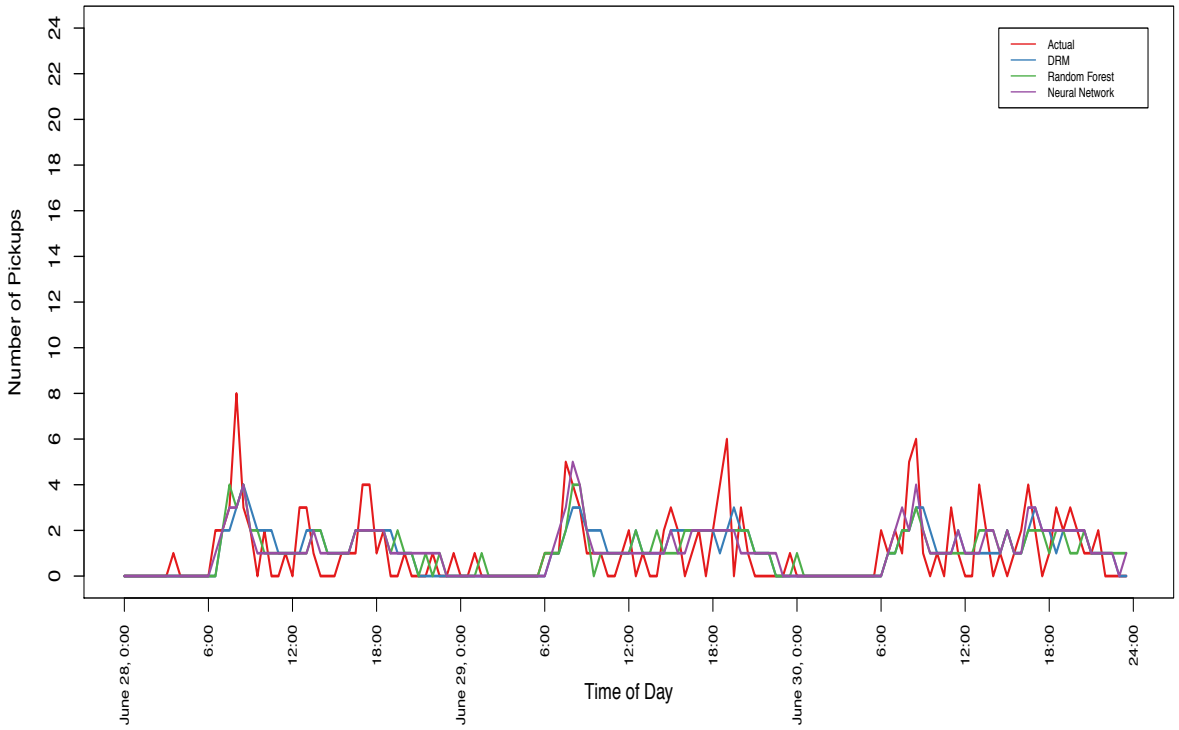


Figure A-5 7th & North Carolina Ave SE the Predicted Number of Pick-ups Versus Actual Numbers

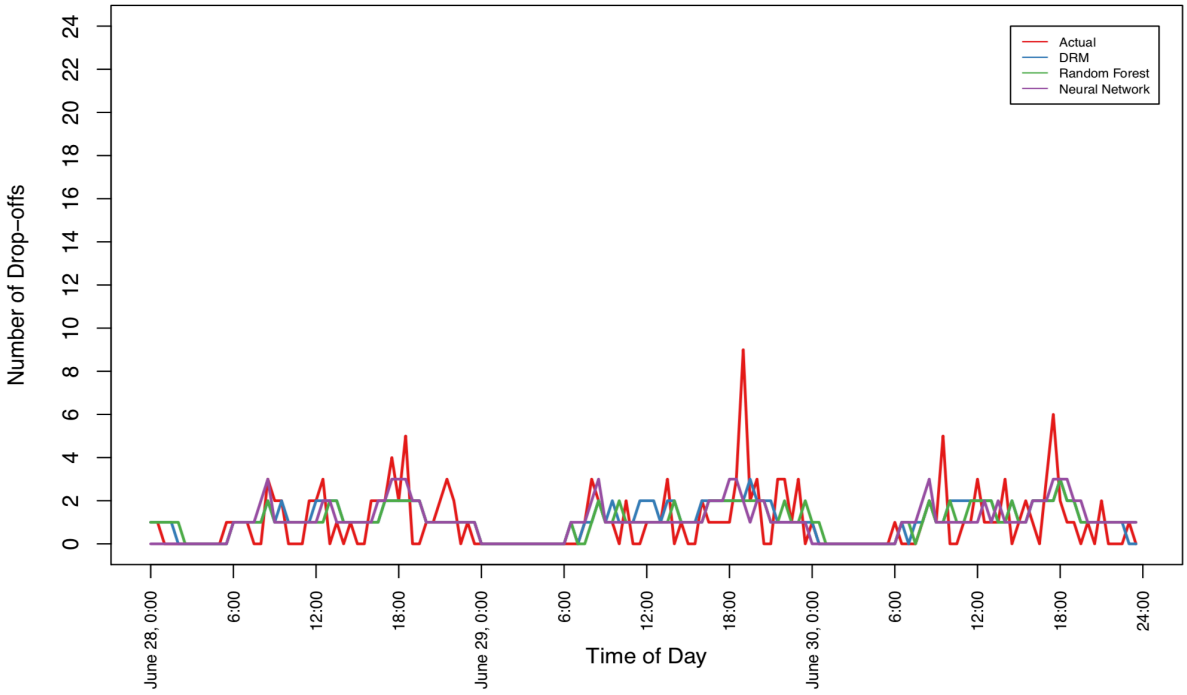


Figure A-6 7th & North Carolina Ave SE the Predicted Number of Drop-offs Versus Actual Numbers

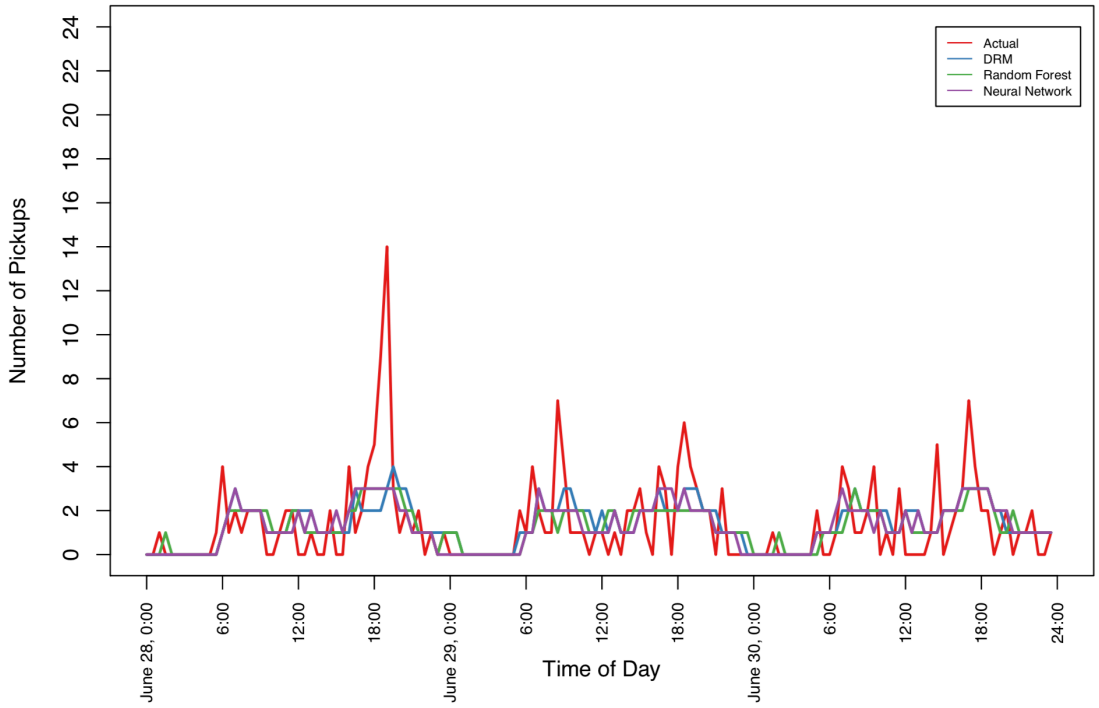


Figure A-7 4th & East Capitol St NE the Predicted Number of Pick-ups Versus Actual Numbers

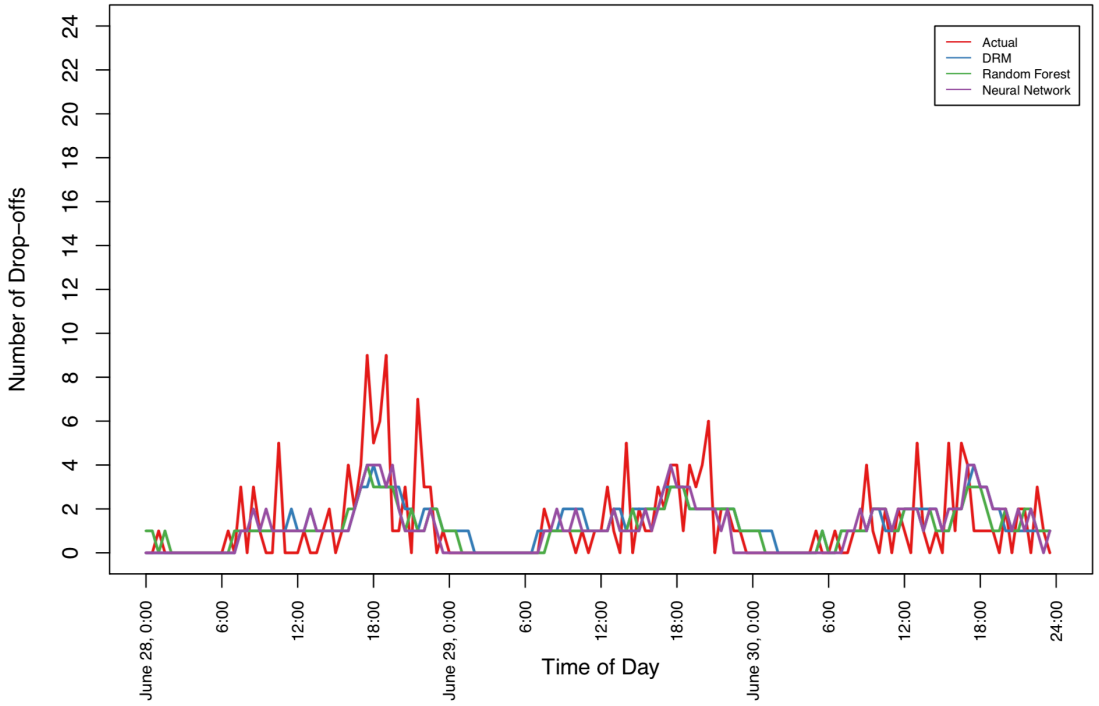


Figure A-8 4th & East Capitol St NE the Predicted Number of Drop-offs Versus Actual Numbers

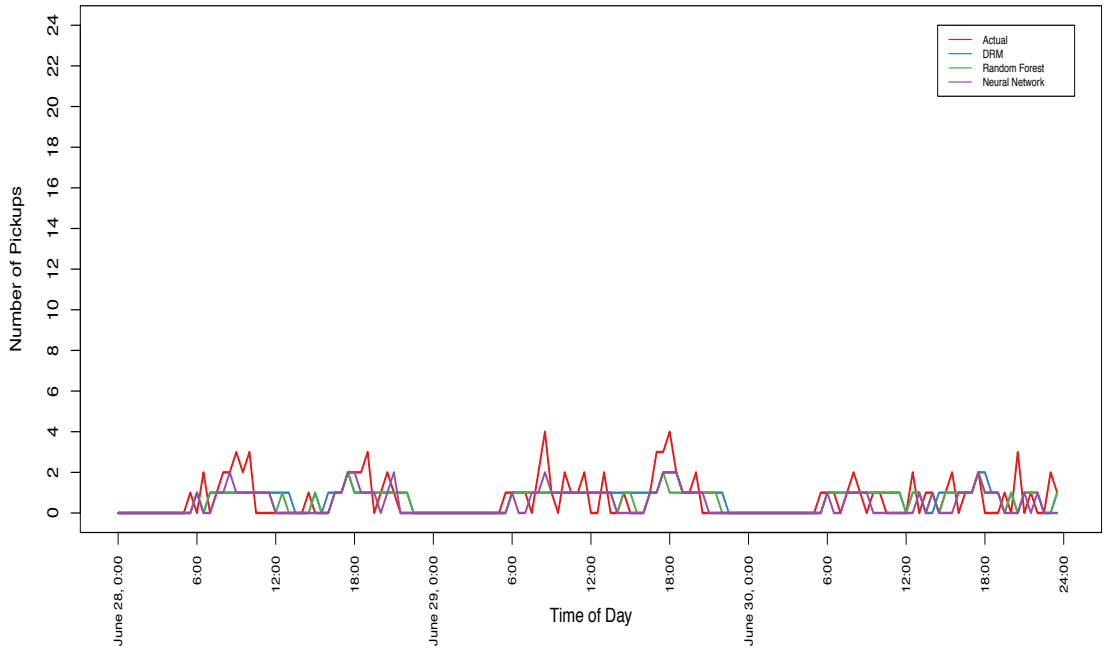


Figure A-9 19th & East Capitol St SE the Predicted Number of Pick-ups Versus Actual Numbers

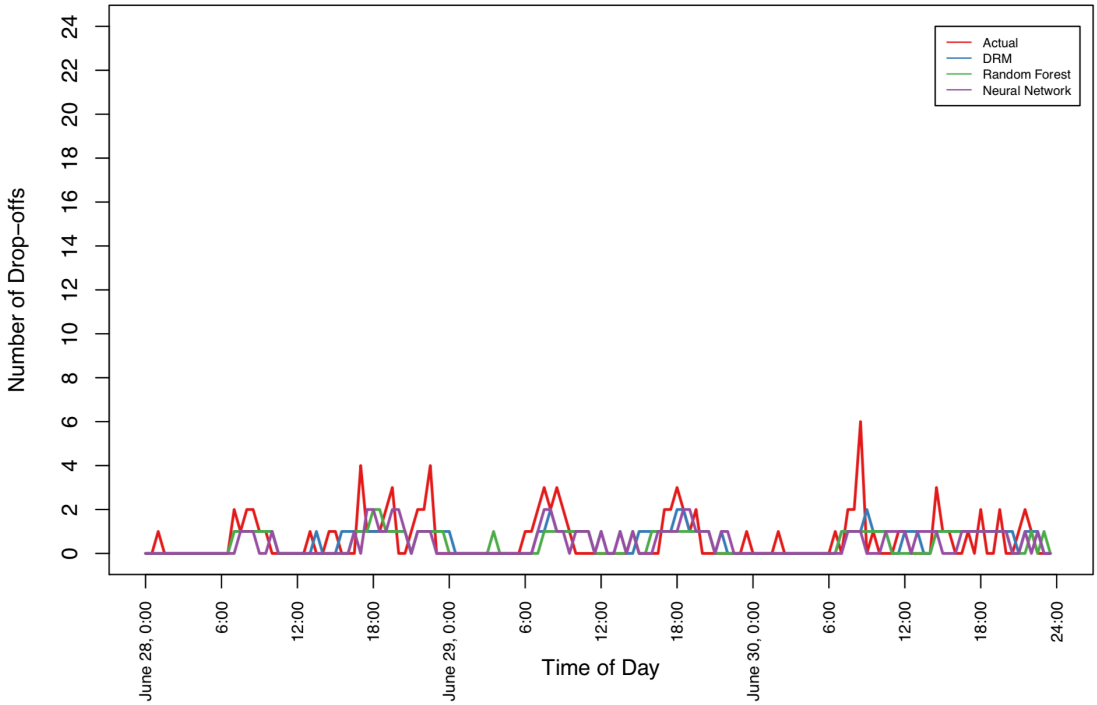


Figure A-10 19th & East Capitol St SE the Predicted Number of Drop-offs Versus Actual Numbers

Table A-1 and Table A-2 show the performance measurements, including mean error, mean absolute error, and root mean squared error of the five methods. It should be noted that since the numbers that are predicted by the models are continuous, as stated before, they are rounded to integer values. As expected, generally, DRM, RF, and DNN have fewer errors (i.e., higher accuracy) compared to the baseline naïve and the naïve seasonal methods. Looking at Table A-1, we can see that the DNN has a smaller root mean squared error on the test sets in seven out of the eight cases in high-demand seasons. The mean absolute error for predicting the pick-ups and drop-offs is around 2 for the high usage station, around 1 for medium usage stations, and less than 1 for the low usage station.

Table A-1 Prediction Accuracy Measurements for the First Test Set

			Accuracy Measures	Method				
				Naïve method	**Seasonal naïve method**	**DRM model**	**Random forest**	**Neural network model**
Jefferson Dr & 14th St SW	Drop-offs	Pick-ups	Mean error	0.02	0.06	-0.09	0.31	-0.03
			Root mean squared error	5.23	4.64	3.40	3.40	**3.30**
			Mean absolute error	3.56	2.85	2.27	2.28	2.08
		Drop-offs	Mean error	-0.06	0.08	-0.10	0.43	-0.19
			Root mean squared error	4.29	3.77	3.41	3.44	**3.22**
			Mean absolute error	2.94	2.54	2.33	2.46	2.19
7th & North Carolina Ave SE	Pick-ups	Pick-ups	Mean error	0.02	-0.38	-0.01	0.01	0.03
			Root mean squared error	1.55	1.54	1.24	1.22	**1.13**
			Mean absolute error	0.94	1.04	0.81	0.81	0.74
		Drop-offs	Mean error	-0.04	-0.27	-0.09	0.04	0.06
			Root mean squared error	1.38	1.42	1.24	1.25	**1.20**
			Mean absolute error	0.88	0.98	0.80	0.81	0.78
4th & East Capitol St NE	Drop-offs	Pick-ups	Mean error	0.02	0.21	0.01	-0.11	-0.10
			Root mean squared error	2.29	1.83	1.71	1.70	**1.64**
			Mean absolute error	1.35	1.00	1.06	1.03	0.97
		Drop-offs	Mean error	0.02	0.31	0.03	-0.14	-0.15
			Root mean squared error	2.56	1.88	1.62	1.54	**1.52**
			Mean absolute error	1.69	1.19	1.10	1.00	0.95
19th & East Capitol St SE	Pick-ups	Pick-ups	Mean error	0.00	-0.23	0.01	-0.08	-0.21
			Root mean squared error	0.94	2.25	0.88	0.85	**0.82**
			Mean absolute error	0.54	0.85	0.58	0.52	0.49
		Drop-offs	Mean error	-0.02	-0.10	0.08	-0.19	-0.18
			Root mean squared error	1.23	1.66	**0.94**	**0.94**	0.96
			Mean absolute error	0.69	0.85	0.60	0.56	0.58

Table A-2 Prediction Accuracy Measurements for the Second Test Set

		Accuracy Measures	Method: Naïve method	Seasonal naïve method	DRM model	Random forest	Neural network model
19th & East Capitol St SE	Drop-offs	Mean error	0.00	-0.64	-0.43	0.31	0.54
		Root mean squared error	1.42	2.15	**1.41**	1.45	1.69
		Mean absolute error	0.61	1.00	0.71	0.79	0.93
	Pick-ups	Mean error	0.00	-0.62	-0.32	0.26	0.68
		Root mean squared error	1.70	2.46	1.48	**1.36**	2.44
		Mean absolute error	0.82	1.22	0.75	0.81	1.29
4th & East Capitol St NE	Drop-offs	Mean error	0.00	-0.58	-0.10	0.19	0.19
		Root mean squared error	0.83	1.68	**0.71**	0.72	0.80
		Mean absolute error	0.38	0.88	0.39	0.42	0.37
	Pick-ups	Mean error	0.00	-0.63	-0.10	0.21	0.24
		Root mean squared error	0.67	1.67	**0.56**	0.70	0.76
		Mean absolute error	0.28	0.83	0.27	0.39	0.39
7th & North Carolina Ave SE	Drop-offs	Mean error	0.00	-0.41	-0.10	0.26	0.18
		Root mean squared error	0.83	1.22	**0.72**	0.75	0.75
		Mean absolute error	0.42	0.69	0.42	0.52	0.44
	Pick-ups	Mean error	0.00	-0.37	-0.13	0.19	0.17
		Root mean squared error	0.79	1.13	0.75	0.73	**0.70**
		Mean absolute error	0.29	0.56	0.38	0.41	0.34
Jefferson Dr & 14th St SW	Drop-offs	Mean error	0.00	-0.17	0.00	0.01	0.02
		Root mean squared error	0.33	0.57	0.33	**0.32**	0.34
		Mean absolute error	0.11	0.26	0.11	0.10	0.12
	Pick-ups	Mean error	0.00	-0.13	-0.01	-0.05	-0.07
		Root mean squared error	0.42	0.58	0.45	0.40	**0.37**
		Mean absolute error	0.15	0.26	0.17	0.13	0.11

Appendix B Hyperparameter Tuning and Model Selection for DNN

B.1 Overview

Hyperparameter tuning is performed on the following variables: number of layers, number of neurons, and dropout rate. Hyperparameter tuning is done on the base model (model which only uses following variables: past observations, weather data, time of the day, day of the week, and monthly indicators.) and the selected hyperparameters are used for training the base model and all other models (models that include other information including the status of stations and membership information.) For a well-tuned model, the same method can be used for the hyperparameter selection of other models.

The search space for the number of layers is {2, 3, 4} (i.e., no hidden layers, one hidden layer, or two hidden layers), and the search space for the number of neurons per hidden layer is {25, 50, 100, 200, 250}. Neural network architecture is picked so that there is no bottleneck layer in the neural network, meaning there is no layer that has a lower number of neurons compared to its previous and succeeding layer.

The dropout rate refers to the rate of eliminating some number of neurons of the hidden layers randomly during training. This method is used to reduce over-fitting and improve generalization error. Search space for dropout rate is {0.4, 0.5, 0.6}. The rate can be seen as the probability of the neuron becoming inactive during training. At test time, dropout is disabled.

The sensitivity of the evaluation metric used is also tested with respect to the number of past observations chosen as an input to the model to find the optimal number of previous observations. Although it is possible to put a high number of past observations

as input and let the neural network decide which ones to choose, this step is done to reduce the training time of the models (Number of epochs for training) and reduce model complexity and over-fitting. The Search space (i.e., number) of past observations is in {1, 2, 4, 6, 12, 24, 48}.

B.2 Training and Validation Dataset for Hyperparameter Tuning

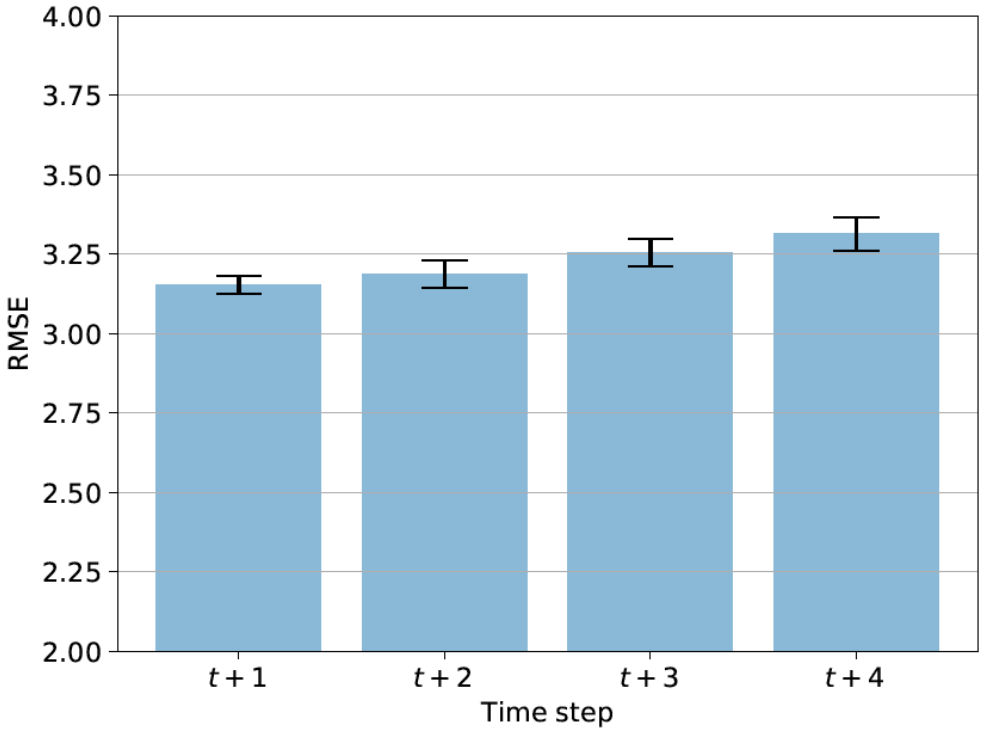
We randomly choose 70% of the 2016 trip data of each station to train the models and use the rest (30%) for comparing different hyperparameters (validation dataset). The 2017 trip data is used for testing.

In the first step, 10 random combinations of hyperparameters are chosen for 20% of stations from each of the groups defined in before. The model is trained based on these hyperparameters and then the root mean squared error of the 4-step ahead prediction (number of pick-ups or drop-offs in 0-30 minute, 30-60 minute, 60-90 minute, and 90-120 minute) on the testing dataset is used as the measurement for comparison. In the second step, the optimal combination of hyperparameters of each station is used for all other stations. In the last step, the combination that has the lowest average root mean squared error among all the stations of that group is selected as the optimal hyperparameters for the corresponding group.

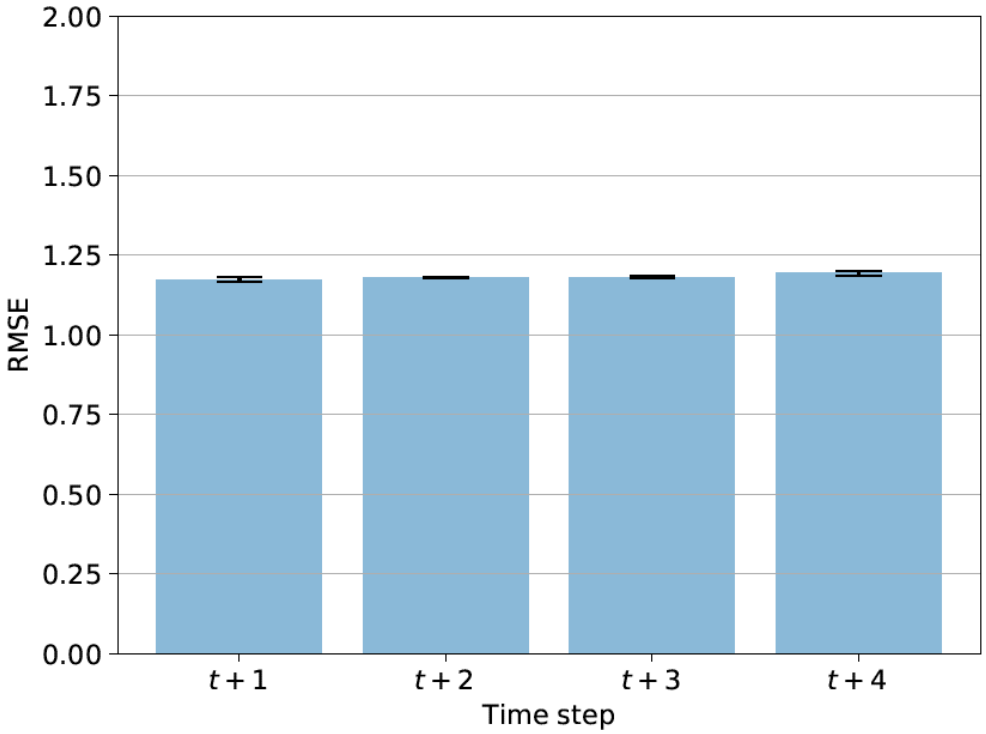
Based on this method, a DNN with 2, 3, and 4 layers shows the lowest average minimum square error for low, medium, and high usage stations, respectively. For the list of all the hyperparameters picked by the method, see Table 5-1.

Figure B-1 shows an example of mean and variance of root mean squared error (whisks) at different time steps for the number of pick-ups of a) high demand station b) medium demand station c) low demand station using the hyperparameter

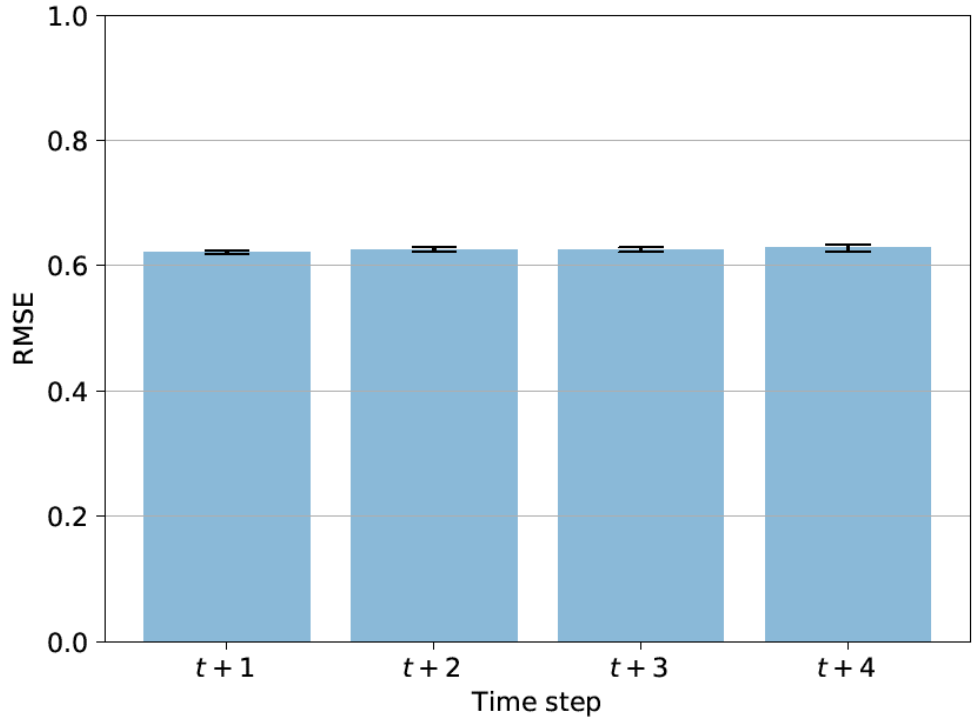
combinations picked by the random search method described above. This visualization and similar visualizations in appendix B are developed by applying the trained models on 2017 trip data (testing dataset). Each model is run 5 times with different initializations to get the variance (whisks). This applies to all the plots with whisks.



a) High demand station



b) Medium demand station



c) Low demand station

Figure B-1 Mean and Variance of Root Mean Squared Error for the Number of Pick-ups a) a High Demand Station (Terminal Number 31247 -Jefferson Dr & 14th St SW) b) a Medium Demand Station (Terminal Number 31610-7th & North Carolina

Ave SE) c) a Low Demand Station (Terminal Number 31516- Rhode Island Ave Metro)

In almost all the instances, root mean squared error monotonically increases for the future steps. Differences between time steps are clearer in high usage stations. This could indicate that the past few observations are not an important factor for predicting pick-ups and drop-offs of low and medium usage stations compared to high usage stations.

B.3 Effect of the Dropout rate

Figure B-2 shows the effect of the dropout rate on the mean and variance of root mean squared error at different time steps for the number of drop-offs of a high demand station.

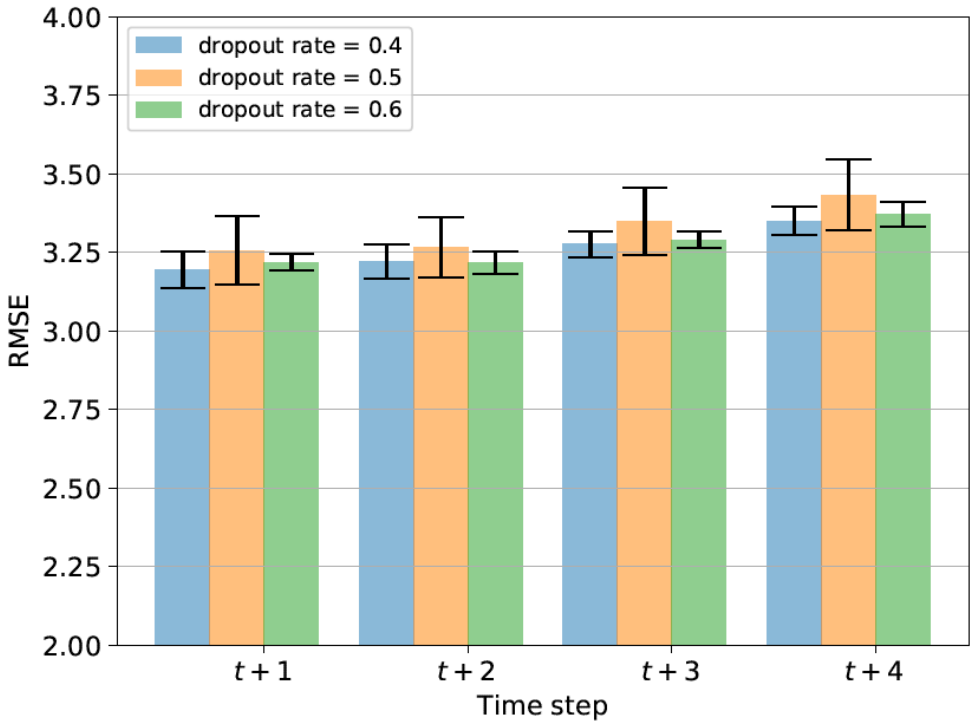


Figure B-2 Effect of Dropout Rate on Root Mean Squared Error for the Number of Drop-offs of a High Demand Station (Terminal Number 31247 -Jefferson Dr & 14th St SW)

As can be seen in Figure B-2, an appropriate value for the dropout rate can increase generalization. However, if the dropout rate takes on large values, then the model will underfit, and the performance drops.

B.4 Effect of the Number of Previous Observations on the Accuracy

Figure B-3 shows the effect of the number of previous observations on the mean and variance of the root mean squared error of the validation dataset for the number of pick-ups of a high demand station (terminal number 31247 -Jefferson Dr & 14th St SW.)

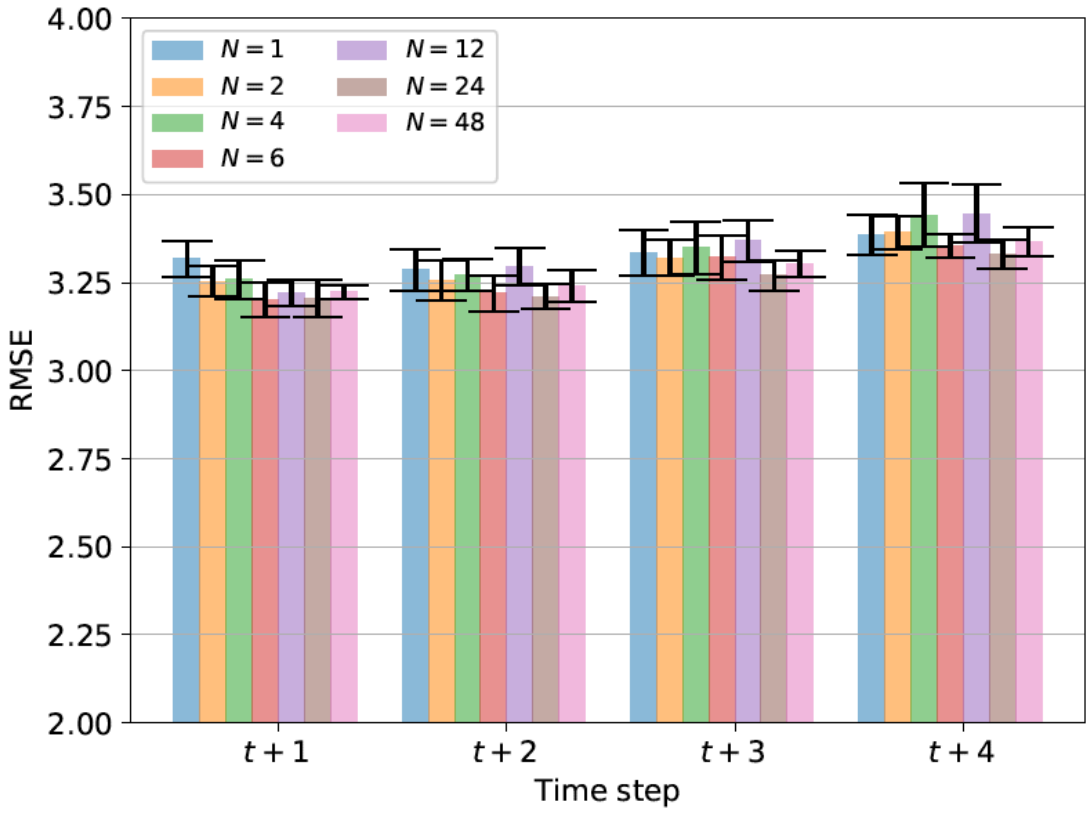


Figure B-3 Effect of Number of Previous Observations on Root Mean Squared Error for the Number of Drop-offs of a High Demand Station (Terminal Number 31247 - Jefferson Dr & 14th St SW)

B.5 Optimizer, Learning-rate, and Training Epochs

Adam optimizer is used as the optimization algorithm for all the groups with learning rate =0.01, beta_1 = 0.9, beta_2 = 0.999, and decay = 1e-4. The number of epochs is set to 30 for all the instances.

References

Amit, Y., & Geman, D. (1997). Shape quantization and recognition with randomized trees. *Neural computation*, *9*(7), 1545-1588.

Anily, S., & Hassin, R. (1992). The swapping problem. *Networks, 22* (4), 419-433.

Benchimol, M., Benchimol, P., Chappert, B., De La Taille, A., Laroche, F., Meunier, F., & Robinet, L. (2011). Balancing the stations of a self service "bike hire" system. *RAIRO-Operations Research*, *45*(1), 37-61.

Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, *5*(2), 157-166.

Breiman, L. (1996). Bagging predictors. *Machine learning, 24* (2), 123-140.

Breiman, L. F. (1984). Classification and regression trees. *CRC press*.

Breiman, L. (2001). Random forests. *Machine learning, 45* (1), 5-32.

Buck, D., Buehler, R., Happ, P., Rawls, B., Chung, P., & Borecki, N. (2013). Are bikeshare users different from regular cyclists? A first look at short-term users, annual members, and area cyclists in the Washington, DC, region. Transportation research record, 2387(1), 112-119.

Cabitracker Website, http://cabitracker.com

Capital Bikeshare Website, https://capitalbikeshare.com

Chalasani, P., & Motwani, R. (1999). Approximating capacitated routing and delivery problems. *SIAM Journal on Computing, 28* (6), 2133-2149.

Chemla, D. M., Meunier, F., & Calvo, R. W. (2013). Bike sharing systems: Solving the static rebalancing problem. *Discrete Optimization , 10* (2), 120-146.

Chemla, D. M., Meunier, F., Pradeau, T., Calvo, R. W., & Yahiaoui, H. (2013). *Self-service bike sharing systems: simulation, repositioning, pricing.* Centre d'Enseignement et de Recherche en Mathématiques et Calcul Scientifique (CERMICS).

Chen, B., Pinelli, F., Sinn, M., Botea, A., & Calabrese, F. (2013). Uncertainty in urban mobility: Predicting waiting times for shared bicycles and parking lots. *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)* (pp. 53-58). IEEE.

Contardo, C. M. (2012). *Balancing a dynamic public bike-sharing system.* Cirrelt.

Cordeau, J. F. (2006). A branch-and-cut algorithm for the dial-a-ride problem. *Operations Research, 54* (3), 573-586.

Datner, S., Raviv, T., Tzur, M., & Chemla, D. (2019). Setting inventory levels in a bike sharing network. *Transportation Science*, *53*(1), 62-76.

Dell'Amico, M., Hadjicostantinou, E., Iris, M., & Novellani, S. (2014). The bike sharing rebalancing problem: mathematical formulations and benchmark instances. *Omega, 45*, 7-19.

Dell'Amico, M., Iori, M. N., & Stutzle, T. (2016). A destroy and repair algorithm for the bike sharing rebalancing problem. *Computers and operations research, 71*, 149-162.

DeMaio, P. (2009). Bike-sharing: History, impacts, models of provision, and future. *Journal of Public Transportation, 12* (4), 41-56.

DeMaio, P., & Meddin, R. (2016). *The bike-sharing world map*. Retrieved from http://www.metrobike.net/

Dietterich, T. (1998). An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting and randomization. *Machine Learning*, 1-22.

Dupond, S. (2019). A thorough review on the current advance of neural network structures. *Annual Reviews in Control*, *14*, 200-230.

Erdoğan, G., Battarra, M., & Calvo, R. W. (2015). An exact algorithm for the static rebalancing problem arising in bicycle sharing systems. *European journal of operational research, 245* (3), 667-679.

Erdoğan, G., Laporte, G., & Calvo, R. W. (2014). The static bicycle relocation problem with demand intervals. *European Journal of Operational Research, 238* (2), 451-457.

Etienne, C., & Latifa, O. (2014). Model-based count series clustering for bike sharing system usage mining: a case study with the Vélib'system of Paris. *ACM Transactions on Intelligent Systems and Technology (TIST)*.

Faghih-Imani, A., & Eluru, N. (2016). Incorporating the impact of spatio-temporal interactions on bicycle sharing system demand: A case study of New York CitiBike system. *Journal of Transport Geography, 54*, 218-227.

Faghih-Imani, A., Eluru, N., El-Geneidy, A. M., Rabbat, M., & Haq, U. (2014). How land-use and urban form impact bicycle flows: evidence from the bicycle-sharing system (BIXI) in Montreal. *Journal of Transport Geography, 41*, 306-314.

Fishman, E. (2016). Bikeshare: A review of recent literature. *Transport Reviews, 36* (1), 92-113.

Fishman, E., Washington, S., & Haworth, N. (2014). Bike share's impact on car use: Evidence from the United States, Great Britain, and Australia. *Transportation Research Part D: Transport and Environment*, *31*, 13-20.

Fishman, E., Washington, S., & Haworth, N. (2013). Bike share: a synthesis of the literature. *Transport Reviews, 33* (2), 148-165.

Forma, I. A., Raviv, T., & Tzur, M. (2015). A 3-step math heuristic for the static repositioning problem in bike-sharing systems. *Transportation research part B: methodological, 71*, 230-247.

Freund, Y., & Schapire, R. E. (1995, March). A desicion-theoretic generalization of on-line learning and an application to boosting. In *European conference on computational learning theory* (pp. 23-37). Springer, Berlin, Heidelberg.

Froehlich, J., Neumann, J., & Oliver, N. (2009). Sensing and Predicting the Pulse of the City through Shared Bicycling. (pp. 1420-1426). Proceedings of the Twenty-First International Joint Conference on Artificial Intelligence.

Ghosh, S., Trick, M., & Varakantham, P. (2016). Robust repositioning to counter unpredictable demand in bike sharing systems. *Twenty-fifth international joint conference on artificial intelligence*, (pp. 3096-3102).

Ghosh, S., Varakantham, P., Adulyasak, Y., & Jaillet, P. (2017). Dynamic repositioning to reduce lost demand in bike sharing systems. *Journal of artificial intelligence research*, 387-430.

Giot, R., & Cherrier, R. (2014). Predicting bikeshare system usage up to one day ahead. *2014 IEEE symposium on Computational intelligence in vehicles and transportation systems (CIVTS)* (pp. 22-29). IEEE.

Hamari, J. S. (2015). The sharing economy: why people participate in collaborative consumption. *Journal of the Association for Information Science and Technology*, 2047-2059.

Hernández-Pérez, H., & Salazar-González, J. J. (2007). The one-commodity pickup-and-delivery traveling salesman problem: Inequalities and algorithms. *Networks: An International Journal*, *50*(4), 258-272.

Ho, S. C., & Szeto, W. Y. (2014). Solving a static repositioning problem in bike-sharing systems using iterated tabu search. *Transportation Research Part E: Logistics and Transportation Review*, 180-198.

Ho, T. K. (1998). The random subspace method for constructing decision forests. *IEEE transactions on pattern analysis and machine intelligence, 20* (8), 832-844.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735-1780.

Kaltenbrunner, A., Meza, R., Grivolla, J., Codina, J., & Banchs, R. (2010). Urban cycles and mobility patterns: Exploring and predicting trends in a bicycle-based public transport system. *Pervasive and Mobile Computing, 6* (4), 455-466.

Kaspi, M. R. (2016). Bike-sharing systems: user dissatisfaction in the presence of unusable bicycles. *IISE transactions*, 1-15.

Kloimüllner, C. P. (2014). Balancing bicycle sharing systems: an approach for the dynamic case. *European Conference on Evolutionary Computation in Combinatorial Optimization*, (pp. 73-84).

Li, Y., Zheng, Y., Zhang, H., & Chen, L. (2015). Traffic prediction in a bike-sharing system. *23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems.* ACM.

Liu, J., Sun, L., Chen, W., & Xiong, H. (2016). Rebalancing bike sharing systems: A multi-source data smart optimization. *22nd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 1005-1014). ACM.

Loh, W. Y. (2011). Classification and regression trees. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 1* (1), 14-23.

McCulloch, W. S. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics, 5* (4), 115-133.

Meisel, S. M. (2010). Synergies of operation research and data mining. *European journal of operational research, 206* (1), 1-10.

Nair, R. (2010). Design and analysis of vehicle sharing programs: a systems approach. *Ph.D. dissertation*, 1-141.

O'Mahony, E., & Shmoys, D. B. (2015). Data analysis and optimization for(Citi) bike sharing. *AAAI*, (pp. 687-694).

Papazek, P., Raidl, G. R., Rainer-Harbach, M., & Hu, B. (2013). A PILOT/VND/GRASP hybrid for the static balancing of public bicycle sharing systems. *International Conference on Computer Aided Systems Theory* (pp. 372-379). Springer Berlin Heidelberg.

Pfrommer, J. W. (2014). Dynamic vehicle redistribution and online price incentives in shared mobility systems. *IEEE Transactions on Intelligent Transportation Systems, 15* (4), 1567-1578.

Raidl, G. R., Hu, B., Rainer-Harbach, M., & Papazek, P. (2013). Balancing bicycle sharing systems: Improving a VNS by efficiently determining optimal loading operations. *International Workshop on Hybrid Metaheuristics* (pp. 130-143). Springer Berlin Heidelberg.

Rainer-Harbach, M., Papazek, P., Hu, B., & Raidl, G. R. (2013). Balancing bicycle sharing systems: A variable neighborhood search approach. *European Conference on Evolutionary Computation in Combinatorial Optimization, lecture notes in computer science*, *7832*, pp. 121-132.

Raviv, T., & Kolka, O. (2013). Optimal inventory management of a bike-sharing station. *IIE Transactions, 45* (10), 1077-1093.

Raviv, T., Tzur, M., & Forma, I. A. (2013). Static repositioning in a bike-sharing system: models and solution approaches. *EURO journal o transportation and logistics, 2* (3), 187-229.

Reiss, S., & Bogenberger, K. (2015, September). Gps-data analysis of Munich's free-floating bike sharing system and application of an operator-based relocation strategy. In *2015 IEEE 18th International Conference on Intelligent Transportation Systems* (pp. 584-589). IEEE.

Safavian, S. R. (1991). A survey of decision tree classifier methodology. *IEEE transactions on systems, man, and cybernetics, 21(3), 21* (3), 660-674.

Saltzman, R. B. (2016). Simulating a more efficient bike share system. *Journal of supply chain and operations management, 14* (2), 36-47.

Schapire, R. E. (1990). The strength of weak learnability. *Machine learning, 5* (2), 197-227.

Schuijbroek, J., Hampshire, R., & van Hoeve, W. J. (2013). Inventory rebalancing and vehicle routing in bike sharing systems.

Shaheen, S. C. (2013). Public bikesharing in North America: early operator understanding and emerging trends. *Transportation research record: journal of the transportation research board*, 83-92.

Shaheen, S., Guzman, S., & Zhang, H. (2010). Bikesharing in Europe, the Americas, and Asia: past, present, and future. *2143*, 159-167.

Shu, J., Chou, M. C., Liu, Q., Teo, C. P., & Wang, I. L. (2013). Models for effective deployment and redistribution of bicycles within public bicycle-sharing systems. *Operations Research, 61* (6), 1346-1359.

Toole design group, LLC and Foursqure ITP. (2013). Philadelphia bike share strategic business plan. *Report*, 1-66.

Vogel, P., & Mattfeld, D. C. (2011). Strategic and operational planning of bike-sharing systems by data mining–a case study. *International Conference on Computational Logistics* (pp. 127-141). Berlin, Heidelberg: Springer.

Vogel, P., Saavedra, B. A., & Mattfeld, D. C. (2014). A hybrid metaheuristic to solve the resource allocation problem in bike sharing systems. *International Workshop on Hybrid Metaheuristics*, (pp. 16-29).

Wang, T. (2014). *Solving dynamic repositioning problem for bicycle sharing systems: model, heuristics, and decomposition* (Doctoral dissertation).

Wang, W. (2016). *Forecasting Bike Rental Demand Using New York Citi Bike Data.* Dublin.

Whittle, P. (1951). *Hypothesis testing in time series analysis* (Vol. 4). Almqvist & Wiksells boktr.

Wolpert, D. H. (1992). Stacked generalization. *Neural networks, 5* (2), 241-259.

Yin, Y. C., Lee, C. S., & Wong, Y. P. (2012). *Demand Prediction of Bicycle Sharing Systems.*

Yoon, J. W., Pinelli, F., & Calabrese, F. (2012). Cityride: a predictive bike sharing journey advisor. *2012 IEEE 13th International Conference on Mobile Data Management* (pp. 3016-311). IEEE.

Yu, B., Yin, H., & Zhu, Z. (2017). Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *arXiv preprint arXiv:1709.04875*.

Zamir, K. R., Bondarenko, I., Burnett, S. T., Brodie, S., & Lucas, K. (2019). Comparative Analysis of User Behavior of Dock-Based and Dockless

Bikeshare and Scootershare in Washington, DC, Metropolitan Area. *Transportation Research Board 98th Annual Meeting* (No. 19-03300)

Zhang, S., Tong, H., Xu, J., & Maciejewski, R. (2019). Graph convolutional networks: a comprehensive review. *Computational Social Networks*, *6*(1), 11.